International Journal of Computer Science and Security (IJCSS)

ISSN : 1985-1553

VOLUME 3, ISSUE 3

PUBLICATION FREQUENCY: 6 ISSUES PER YEAR

Editor in Chief Dr. Haralambos Mouratidis

International Journal of Computer Science and Security (IJCSS)

Book: 2009 Volume 3, Issue 3
Publishing Date: 30-06-2009
Proceedings
ISSN (Online): 1985 -1553

IJCSS Journal is a part of CSC Publishers
http://www.cscjournals.org

©IJCSS Journal
Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers

Table of Contents

Volume 3, Issue 3, May/June 2009.

System-Level Modeling of a Network-on-Chip

Ankur Agarwal ankur@cse.fau.edu
Assistant Professor, Department of
Computer Science and Engineering
Florida Atlantic University,
Boca Raton, 33431, Florida

Abstract

This paper presents the system-level modeling and simulation of a concurrent architecture for a customizable and scalable network-on-chip (NoC), using system level tools (Mission Level Designer (MLD)). MLD supports the integration of heterogeneous models of computation, which provide a framework to model various algorithms and activities, while accounting for and exploiting concurrency and synchronization aspects. Our methodology consists of three main phases: system-level concurrency modeling, component-level modeling, and system-level integration. At first, the Finite State Processes (FSP) symbolic language is used to model and analyze the system-level concurrency aspects of the NoC. Then, each component of the NoC is abstracted as a customizable class with parameters and methods, and instances of these classes are used to realize a 4×4 mesh-based NoC within the MLD environment. To illustrate and validate the system-level operation of the NoC, we provide simulation results for various scheduling criteria, injection rates, buffer sizes, and network traffic patterns.

Keywords: Network-on-Chip, System-Level Design, Concurrency Modeling, System Modeling, Models of Computation

1. INTRODUCTION

System complexity, driven by both increasing transistor count and customer need for increasingly savvy applications, has increased so dramatically that system design and integration can no longer be an after-thought. As a consequence, system level design, long the domain of a few expert senior engineers, is coming into its own as a discipline. SystemC, introduced seven years ago, has made it possible to co-design and co-develop software and hardware, and hence, such components [1]. However, integration of an entire product at the SystemC level would take too long to complete. In this paper we propose a methodology to separately address design and optimization at a higher level, viz., Architect's level, where components pre-developed (using SystemC or any other equivalent system language) can be integrated with each other to build subsystems, and ultimately a system, with the (sub) system optimized across several more global QoS parameters. Note that for this to be practically feasible, the components have to be abstracted to a higher level, so system modeling and analysis, and final architecture selection, can be completed rapidly.

A system design process is inherently complex. The design involves multiple representations, multiple (design) groups working on different design phases, and a complex hierarchy of data and applications [2]. The different groups bring different perspectives towards system design. The system or product inconsistencies primarily arise out of lack of appropriate communication among various design teams. For example, the concerns of a hardware design engineer are different from that of a software designer [3]. Such constraints lead to an increase in product development cycle and product development cost, thereby reducing system design productivity [4]. To counter this, one will have to exploit the principle of “design-and-reuse” to its full potential [5]. Then, a system (subsystem) would be composed of reusable sub-systems (components).

The network-on-chip (NoC) architecture can improve this declining design productivity by serving as a reusable communication sub-system for an embedded device [6]. NoC provides a multi-core architecture for managing complexity by incorporating concurrency and synchronization. This NoC architecture may comprise components such as routers, input and output buffers, network interfaces, switches, virtual channel allocators, schedulers and switch allocators [7]. To develop a system from such reusable components, one has to design and develop variants of each component. For example, buffer size is a customizable parameter for an input buffer. Similarly, scheduling criteria provide customizability for schedulers. A system architect estimates performance and quality-of-service (QoS) parameters for various system configurations. Components need to encapsulate their performance metrics for various useful parameter combinations, in order to help the architect make informed decisions. We propose that a system be modeled in advance of the architect’s design phase. Such a model is analyzed to ensure system functionality at an abstract level. This model can then be ported to the architect’s design phase, for analyzing the performance of a system and for estimating the resources needed for mapping an application onto the system. This model must allow one to manipulate a set of parameters to fine tune system performance. Such a system model needs to have a high level representation of various performance and QoS parameters for subsystems and components. These performance and QoS parameters can then be traded-off against each other in the system model, to yield a global optimization. Such a model at the design phase will allow one to make key decisions and reduce the scope of the multidimensional search space. These key decisions may include the number of processors, hardware-software partitioning, estimated performance values for new components, and the use of existing components in software or hardware. Such design decisions have the potential to significantly enhance productivity of system design.

Such system modeling, analysis and design will not be an effective solution until we have a mechanism for modeling concurrency and synchronization issues [8, 9]. This requires representation of various activities and algorithms with appropriate MoC [10, 11, 12]. This defines our two goals for system-level designers: (1) To model the system functionality well in advance of building the actual computing system in order to provide a level of flexibility in system design. (2) To be able to manipulate an abstract set of design elements simultaneously to generate different sets of QoS parameters and performance metrics and fine tune the system model.

A model of computation (MoC) is a mathematical formalism that captures and allows us to reason about a computational system or concept independent of its implementation details. Different MoC have evolved to represent the reasoning in different areas of focus [13]. A synchronous local region of a NoC might require one or more such MoC to co-exist and interact. Further, to reduce simulation time and to integrate the subsystems into an integrated system model, other MoC may be needed. Thus, several MoC are needed for modeling an integrated system. In such a system, each component or subsystem of the system should be able to use any allowed MoC and more importantly should retain its behavior after integration of the system. Consider also the case in which a subsystem uses two or more local regions (or islands) of the NoC. These are connected by a switch in a NoC. For example consider a digital camera as a component, or as a subsystem of a much larger system, such as a wireless handheld device (system). It is possible that its design would not fit into one local region. Under such a scenario we should also be able to address the concurrency and synchronization issues because of shared resources (both local and global). We have integrated appropriate MoC to model our NoC architecture.

In this paper, we present a methodology to obtain a system-level model of a NoC which is quality-of-service driven, customizable, and parameterizable. The NoC implementation uses several MoC to model different regions. Due to the distributed nature of the NoC, the modeling of concurrency is essential for a robust design. The contributions of this paper are as follows:

1) We propose a methodology wherein the FSP symbolic language is first used to model and analyze the system-level concurrency aspects of the NoC, before any component-level modeling is done. This prevents the introduction of concurrency issues (such as deadlock and livelock) in the subsequent component-level modeling and integration phases.
2) The MLD environment is used to model each component of the NoC, using different MoC. Each component is abstracted as a customizable class with parameters and methods. Only the system-level functionality of each component is modeled, i.e. no gate-level design is done. Indeed, the goal is to obtain a rapid customizable system-level design, which the system architect can use in iterations until the final design.
3) Instances of these classes of components are used to realize a 4×4 mesh-based NoC, in the integration phase. To illustrate and validate the system-level operation of the NoC, we provide simulation results for various scheduling criteria, injection rates, buffer sizes, and network traffic patterns.
4) The NoC architecture is implemented on a field Programmable gate array (FPGA), and area results are abstracted.

The goal of this paper is not to propose yet another improved NoC architecture, but rather to develop a methodology to accelerate the development of NoC architectures, by performing system-level modeling of the NoC, taking into account concurrency issues. In the process, we show how to apply two complementary tools to the NoC modeling: FSPes and MLD. Once the NoC has been designed using these two tools, it can be easily ported to an FPGA, which allows us to rapidly obtain gate counts and other performance measures for the NoC. Hence, this methodology allows a system architect to rapidly evaluate the performance of a NoC by using system-level modeling, instead of tedious gate-level modeling, while still obtaining gate-level results.

After this Introduction, Section 2 gives an overview of the NoC architecture to be modeled, and discusses the MoC to be used at the system, subsystem and component levels of the NoC. Section 3 presents the system-level concurrency modeling of the NoC architecture using FSP. Section 4 introduces the MLD environment, and details each class of component, along with its set of customizable parameters. Section 5 presents and discusses numerical results for both the simulation of the MLD (in terms of latency) and its emulation on a FPGA (in terms of number of gates). Section 6 concludes the paper.

2. NETWORK-ON-CHIP MODELING USING MoC

To enhance the declining system design productivity one will have to introduce and adopt of new technology, and by addressing system-level design concerns at a higher level of abstraction. A system architect must be able to analyze system performance and complete “what-if-scenarios” from the executable specifications at the initial phases of the design cycle. This will help in reducing and possibly eliminating the re-spins in the design phase, thereby increasing the design productivity. To achieve such capability it is necessary to separate the design concerns, specifically computation from communication. This has led to the exploration of layered system architecture for embedded system development. The concerns related to each layer will be modeled separately, independent of the underlying architectures. This will help us increase the amount of component re-use by facilitating the portability of the components onto the different architectures. These layers are as shown in Figure 1. In this paper we discuss the components associated with the two following layers: the communication backbone and the communication protocol. The components from these two layers have then been designed using system level modeling environment “Mission Level Designer” (MLD).

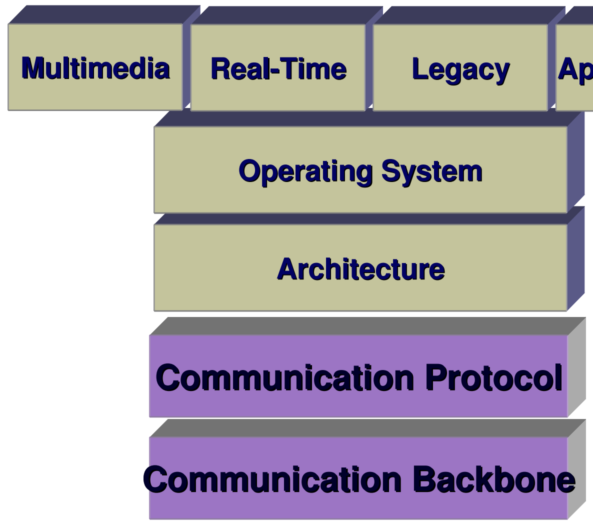


Figure 1: Layered Architecture for System Design.

A NoC is designed as a layered architecture and comprises two main layers: the communication protocol layer and the communication backbone layer. The communication protocol layer consists of the network interface (NI) and is responsible for decoupling communication from computation and packetization/depacketization. Resources such as a general purpose processor (GPP), digital signal processor (DSP), FPGA, application specific integrated circuit (ASIC), memory, or any other hardware element, are connected to the communication backbone layer components through the NI’s. A resource is called a producer (P) when data originates from it, and a consumer (C) when data is received by it. The communication backbone layer, on the other hand, is responsible for buffering, network routing, switching, flow-control, and prioritization of data. It is made up of routers, buffers and links. Figure 2 shows a 3×3 NoC architecture.

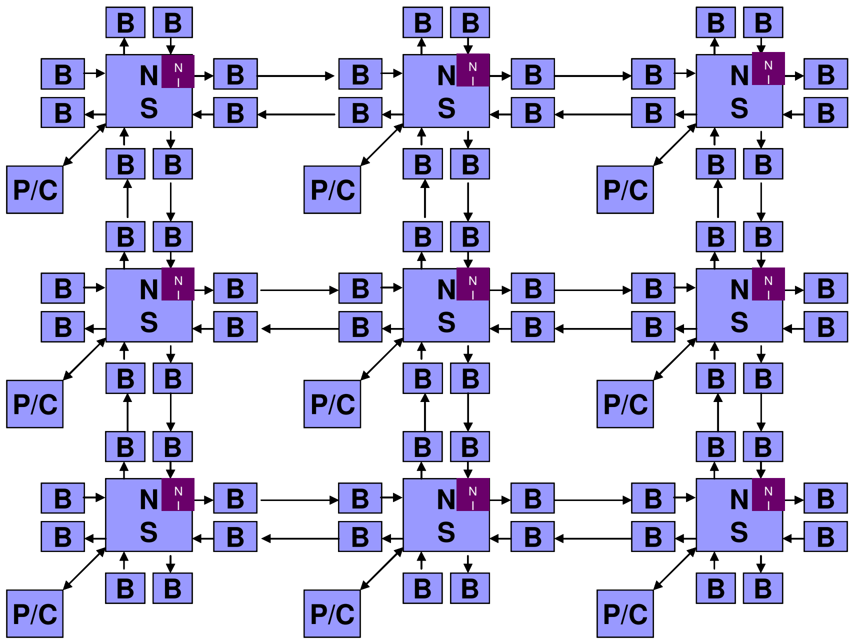


Figure 2: A 3×3 Network-on-Chip Architecture.

The Bi, Bo, P, C, N, S, and NI acronyms denote input buffer, output buffer, producer, consumer, node, scheduler and network interface, respectively. Links provide connections between routers. Routers are connected to each other and to the NI, as per a specific topology. The router block includes the node (N) and scheduler (S) components. The scheduler is responsible for controlling the traffic flow, while the actual data is routed via the node.

At the system (top) level, the NoC model must be defined at a high level of abstraction for it to be useful for performance evaluation [14]. Lower level descriptions, such as register transfer level (RTL) code and source code, while appropriate for the design level, would slow down the trade-off analysis. We should be able to adjust one of these parameters to fine tune system performance and yield different combinations of cost-performance-QoS-power dissipation. One may wish to do this on a dynamic basis for better power management of the system in the field. Thus, we can argue that at the system level, we would need some kind of manager to dynamically optimize the system-level design parameters. The local region for a NoC is again divided into two different domains: NoC at the subsystem level and NoC at the component level. At the sub-system level, we address local issues (as relating to a particular subsystem) rather than global issues. Such a subsystem will usually be some DSP subsystem, IP-core, FPGA, or ASIC. A component is another essential part of the local region. A set of components taken together would constitute the higher subsystem level. At this level, the designer would not have to worry about addressing the system-wide concurrency and synchronization issues. The design should be highly reusable in order to be utilized in other products and scalable in order to be mapped into the higher domains, i.e. the subsystems and the systems. This component level would comprise software components and a computation part, which in turn could be represented by electronic components and computer architecture components.

3. CONCURRENCY MODELING USING FSP

During the previous decade one could enhance the system performance by simply increasing the clock speeds. The International Technology Roadmap of Semiconductors (ITRS) predicts that the saturation point for these clock speeds is nearing [16, 17, 18]. Thus, we need to find other innovative ways of further enhancing performance. One way that could help in this direction is by exploiting concurrency [8]. However, if concurrency issues are not addressed properly, then it may lead any system into a deadlock state. In a multiprocessing environment there are various processes executing simultaneously. For example, in a NoC, at a given time, every resource may either be producing or consuming some data packets. Thus, it can be said that almost every element (node, link, buffer, router, scheduler, resource, etc…) might be communicating with another element at some point perhaps concurrently on the NoC platform. Therefore, there are several inter-processes communications in such a model. If these inter-process communications are not modeled properly then the system may fail. Such a failure may not be detected at the system integration phase. This is due to the fact that these failures are intermittent, which occur only under certain conditions, but nevertheless may be catastrophic. A system may be more prone to intermittent failures if concurrency concerns are not addressed properly. These intermittent failures may finally result into a deadlock or a livelock state. Thus, it is very important to model concurrency issues in such NoC systems.

The main issue is to be able to analyze whether after following all the steps for designing a concurrent system, the new system design will still have any concurrency concerns. There are various MoC which can achieve concurrency [19]. Some of these MoC's include communicating sequential processes (CSP) [20], pi-calculus [21], lambda calculus [22] and finite state machines (FSM) [23]. CSP, pi-calculus and lambda-calculus offer an effective mechanism of specification and verification. Pi-calculus is a well-defined process algebra that can be used to describe and analyze process systems. It allows mobility of communication channels, includes an operational semantics, and can be extended to a higher-order calculus, where not only channels but whole processes can be transferred. However, they are based on mathematical models; they are not only hard to understand for a software/hardware designer but also hard to apply practically.

FSM's have been extensively used in the past, especially in the domain of modeling of control applications. But these FSM's are not able to address the increasing size of the software content in the systems. Hierarchical FSM's have replaced FSM's in modeling concurrent applications. Several tools and methodologies have also been introduced which address these concurrency concerns. Unified Modeling Language (UML) addresses these concurrency concerns in the form of state diagrams and sequence charts [24]. But UML may not be useful for analyzing the system exhaustively for concurrency failures.

FSP and Labeled Transition System Analyzer (LTSA) provide a framework for modeling concurrency and analyzing it exhaustively [25]. FSP is a language based on CSP. But unlike CSP a system designer will not have to analyze the system specification in the form of a mathematical model in order to expose concurrency issues in the system. We can develop a simplified concurrency model in FSP and analyze the concurrency issues with LTSA graphically. LTSA also provides a capability of exhaustive analysis to uncover synchronization issues such as deadlocks and livelocks.

3.1 Concurrency Model for NoC

To develop a concurrency model, we first write a high-level specification. This should not be platform or data dependent. We then identify the concurrent processes in our model. As the concurrency issues arise only among interacting processes, not in the (sequential) internal data processing, we model only the (external) interaction among various processes. This reduces the model complexity significantly; models execute faster due to reduced state exploration. Figure 3 shows a high-level concurrency modeling flow diagram.

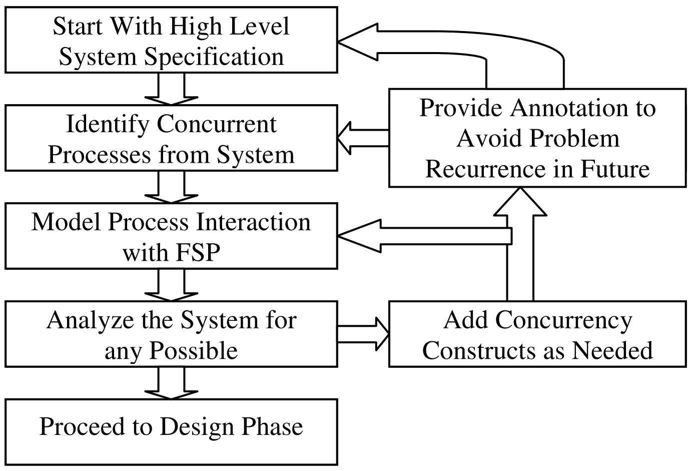

Start With High Level System Specification
Identify Concurrent Processes from System
Model Process Interaction with FSP
Analyze the System for any Possible
Proceed to Design Phase
Provide Annotation to Avoid Problem Recurrence in Future
Add Concurrency Constructs as Needed

Figure 3: Concurrency Modeling Flowchart.

To model concurrency, we start with a very high-level specification. This high-level specification must not contain any internal details of the process behavior. It encompasses the following:
* Data will be received in serialized packet format;
* There will be several paths available arranged in a matrix fashion for the data packet to travel;
* Data may be buffered at each intersection;
* Further routing is available based on the availability and congestion of the links at the destination;
* Packets will contain destination addresses and priorities;
* Links may be unidirectional (one link for each direction) or bi-directional.

We further made two assumptions to simplify our concurrency model: links are unidirectional and nodes are arranged in a 2-D mesh. Concurrent processes in a system can be identified from these high-level specifications. In the NoC model, the identified concurrent processes were links, buffers, schedulers, nodes, producers and consumers. We then define detailed specifications for all individual processes. To model concurrency one must follow an incremental approach. In the NoC model, we first developed the producer and link processes. We then checked for concurrency issues in these two processes before including other processes in the model. Since links are attached to buffers, we then added a buffer process to the model.

As concurrency concerns arise only when two or more processes interact with each other, and not from the internal actions of any processes, we should therefore only model interactions among various processes and not their internal actions. Further, a NoC model with nine nodes and schedulers along with thirty-six buffers, nine producers and consumers and hundreds of links will comprise of tens of thousands of states. Thus it will not only be difficult but almost impossible to model it with FSP and analyze it with LTSA. Therefore, we abstracted this model to represent only one representative scenario of interaction. If this interaction does not have any concurrency issues then other interactions may be replicated in a similar manner to avoid any deadlock or livelock in the system. This abstracted model is represented in Figure 4.

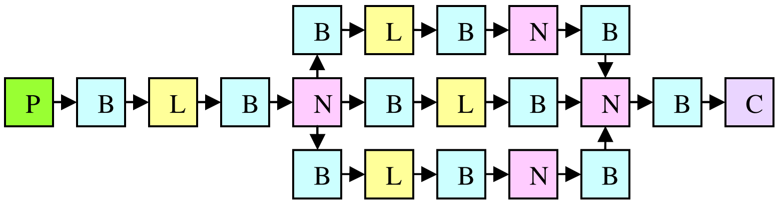


Figure 4: NoC Model for Interaction Between One Producer Process and One Consumer Process.

Further, from the implementation of the link process we realized that a link is responsible for just forwarding the data to the next process. In this it does not play any role which may cause any concurrency concerns. Thus we eliminated the link process from the model. We further reduced this model shown in Figure 3, into a more simplified model by removing unnecessary data paths. Instead of showing three data paths (three buffers connected to one node) we represented the model with two data paths (two buffers with one node). This is due to the fact that synchronization issues will arise when more than one process tries to interact with another process. But as long as the number of similar processes is more than one, we can have a general model to represent these interactions. It may be concluded that the number of buffers (as long as it is more one) will not make any difference in addressing concurrency issues. However, it should be noted that a node with one buffer will not have the same implementation as a node with two buffers. Figure 4 represents a further simplified model with a single data path for the NoC.

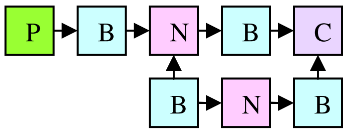


Figure 4: NoC Model for Interaction Between one Producer Process and one Consumer Process with Link Processes Eliminated.

The final abstracted model as shown in Figure 4 has been described in Figure 5. We can realize from the model implementation that there are one producer process - p1, two buffer processes - b1 and b2, one scheduler process – s1. The final composition process includes all of the above processes. These processes have been exhaustively tested. Figure 6 shows results for an exhaustive simulation. It can be seen from simulation results that there are 351 different possible states and 882 possible transitions among these states in the system. None of these states are involved in any deadlock and livelock.

```
PRODUCER = (buffAvail → (hiPriDataOut → PRODUCER | midPriDataOut → PRODUCER)).

const N      = 2
range Data_Range = 0..N
BUFFER = STATE[0][0],
STATE[a:Data_Range][b:Data_Range]
   = (when ((a+b)<N) buffAvail → (hiPriDataIn  → STATE[a+1][b] |midPriDataIn → STATE[a][b+1])
      |when (a>0) shortDataInBuff → nodeGrantFast → hiPriOut → (confirm → STATE[a-1][b]
                                                                |notConfirm → STATE[a][b])
      |when((a==0)&& b>0) dataInBuff → nodeGrantSlow → midPriOut → (confirm → STATE[a][b-1]
                                                                       |notConfirm → STATE[a][b])).
const M = 1
range Node_Requests = 0..M

SCHEDULER = STATE[0][0][0][0],
STATE[h1:Node_Requests][l1:Node_Requests][h2:Node_Requests][l2:Node_Requests]
                    = (when (((h1+l1+h2+l2)<M)) shortDataInBuff1                → STATE[h1+1][l1][h2][l2]
                      |when (((h1+l1+h2+l2)<M)) dataInBuff1                     → STATE[h1][l1+1][h2][l2]
                      |when (((h1+l1+h2+l2)<M)) shortDataInBuff2                → STATE[h1][l1][h2+1][l2]
                      |when (((h1+l1+h2+l2)<M)) dataInBuff2                     → STATE[h1][l1][h2][l2+1]
                      |when (h1>0) nodeGrantFast1 → (outBuffAvail1      → STATE[h1-1][l1][h2][l2]
                                                     |outBuffNotAvail1          → STATE[h1-1][l1][h2][l2])
                      |when (h2>0) nodeGrantFast2 → (outBuffAvail2      → STATE[h1][l1][h2-1][l2]
                                                     |outBuffNotAvail2          → STATE[h1][l1][h2-1][l2])
                      |when ((h1==0)&&(h2==0)&&(l1>0)) nodeGrantSlow1   → (outBuffAvail1
                                                                  → STATE[h1][l1-1][h2][l2]
                                          |outBuffNotAvail1       → STATE[h1][l1-1][h2][l2])
                      |when ((h1==0)&&(h2==0)&&(l2>0)) nodeGrantSlow2   →  (outBuffAvail2
                                                                  → STATE[h2][l1][h2][l2-1]
                                          |outBuffNotAvail2       → STATE[h2][l1][h2][l2-1])).

||FINAL1 = ({b1, b2}:BUFFER||s1:SCHEDULER||p1:PRODUCER)/{
            b1.hiPriDataIn/p1.hiPriDataOut,b1.midPriDataIn/p1.midPriDataOut,b1.buffAvail/p1.buffAvail,
            b1.shortDataInBuff/s1.shortDataInBuff1,b1.dataInBuff/s1.dataInBuff1,
            b1.nodeGrantFast/s1.nodeGrantFast1,b1.nodeGrantSlow/s1.nodeGrantSlow1,
            b2.shortDataInBuff/s1.shortDataInBuff2,b2.dataInBuff/s1.dataInBuff2,
            b2.nodeGrantFast/s1.nodeGrantFast2,b2.nodeGrantSlow/s1.nodeGrantSlow2,
            s1.outBuffAvail1/b1.confirm,s1.outBuffNotAvail1/b1.notConfirm,
            s1.outBuffAvail2/b2.confirm,s1.outBuffNotAvail2/b2.notConfirm}.
```

Figure 5: FSP Implementation for Abstracted NoC Model.

```
Compiled: BUFFER
Compiled: SCHEDULER
Compiled: PRODUCER
Composition:
FINAL1 = b1:BUFFER || b2:BUFFER || s1:SCHEDULER || p1:PRODUCER
State Space:
 24 * 24 * 9 * 2 = 2 ** 15
Composing...
-- States: 351 Transitions: 882 Memory used: 3043K
Composed in 110ms
FINAL1 minimising........
Minimised States: 351 in 46ms
No deadlocks/errors
Progress Check...
-- States: 351 Transitions: 882 Memory used: 3397K
No progress violations detected.
Progress Check in: 31ms
```

Figure 6: Simulation Result for Abstracted NoC Model.

4. COMPONENT MODELING WITH MLD

Our NoC model has been designed in an object oriented manner, wherein each component is an instance of a class with properties and methods. In this section we describe the different classes defined in order to build the NoC model. These classes are (where the capital letters indicate that a class is being referred to): Producer, InputBuffer, Scheduler, Router, OutputBuffer, and Consumer. While being self-contained as per the principles of object-oriented design, these classes need to interact with one another.

The NoC model supports component customization, which eases the task of a design architect. The latter can now change the actual system model to understand the impact of various design parameters without the need for changing the actual design. Thus, it has the potential to provide a more effective analysis of the result by investing less time as compared to traditional performance analysis. The customization parameters are discussed as part of the description for each class. The MLD tool was used to design the classes. The next section provides a brief overview of MLD and the motivations for choosing it for our NoC design.

4.1 System-Level Design with MLD

MLD is a system-level modeling environment which allows one to model a system at an abstract level. MLD supports modeling in different domains such as discrete event (DE), synchronous data flow (SDF), finite state machine (FSM), dynamic data flow (DDF), and synchronous reactive (SR), among others. Multiple domains can be combined to represent a system model. The performance analysis of a model is done at an abstract level, therefore simulations run faster as compared to other modeling environments, which are C, C++ or SystemC-based.

System models are implemented in MLD through either a graphical editor or the Ptolemy Tool command language, PTcl. The functionality of the modules may be specified by a hierarchical block diagram, a finite state machine, a module defined in C/C++ language, or by a PTcl module definition.

4.2 Producer Class

A producer is instantiated from the Producer class. It comprises a resource and a resource network interface. A producer generates the required traffic pattern and packetizes the data into flits. A flit is the smallest unit of communication supported in the NoC. The Producer class has been implemented with a synchronous data flow model of computation as it is responsible for continuous data flow.

Referring to Figure 2, a producer outputs a flit when buffAvail is asserted by the corresponding buffer. A flit is time-stamped at the time of its generation. The timestamp is used to determine the latency involved in delivering the flit. The source and destination address fields of the flit header are updated at this time. The flit header has fields for its priority, timestamp, X-direction of source address, Y-direction of source address, X-direction of destination address, and Y-direction of destination address. The priority of this flit is governed as per a statistical distribution block. For example, in the case of a uniform distribution pattern, every third flit will be a high priority flit. Once the new flit has its timestamp, source and destination addresses and priority fields updated, it is then forwarded to the output through dataOut.

The customizable parameters for the Producer class are: (1) the distribution pattern of the data; (2) the packet injection rate, i.e. the amount of data generated as a function of time; (3) the priorities of the generated data - High, Mid or Low. Each set of parameters is described below. We used three statistically generated traffic distribution patterns – uniform traffic with linear destination, random, and application-specific patterns.

The packet injection rate represents the number of flits per cycle injected into the network for transmission. Defining it as a customizable parameter, allows us to test the NoC model for varying load conditions. The injection rate is changed from 0.1 to 1.0 in equal increments of 0.1 to check the resultant network latency. An injection rate of 1.0 is equivalent to a producer outputting a flit every clock cycle. Similarly, a rate of 0.33 represents the injection of a flit in every three clock cycles.

We provided three priority levels for data packets in our NoC model: High priority, Mid priority and Low priority. High priority supports control signals such as Read (RD), Write (WR), Acknowledge (ACK), and interrupts. Therefore, high priority data is a short packet (single flit packet). Mid priority supports real-time traffic on the system, while Low priority supports non-real time block transfers of data packets. We have defined control signals as High priority because the data must respond immediately to a control signal. Therefore, a control signal must reach its destination in time to manage the communication flow of the network. Real-time data must be delivered in real-time bounds. Therefore, we have assigned Mid Priority to real-time data. The rest of the data on the network belongs to the Low priority class. The number of priority levels is a customizable parameter.

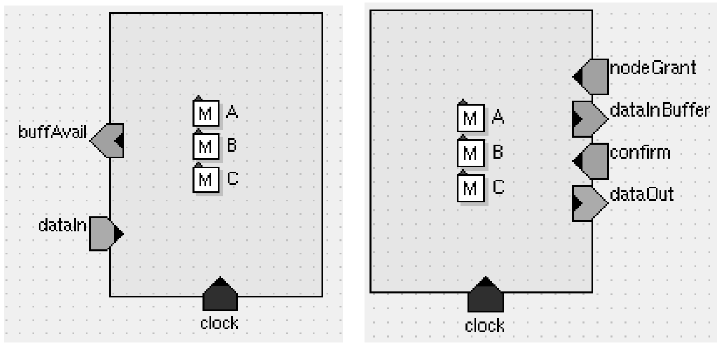
4.3 InputBuffer Class

An input buffer is instantiated from the InputBuffer class. It contains a buffer, a buffer scheduler, and a virtual channel allocator. An input buffer stores the incoming flits, generates the proper handshaking signals to communicate with the scheduler and forwards the flits to the router. An input buffer has an input block and an output block. These two blocks are controlled by a state machine. Thus, we have implemented InputBuffer in the DE and FSM domains. Two concurrent FSM's are responsible for storing the input data at the input terminals of the input buffer and forwarding the data to a router at the output terminal of the input buffer. The DE domain is used for implementing a handshaking protocol. A data forwarding path has been implemented based on a "request-grant" signaling approach (other NoC implementations refer to it as flow control logic). Incoming flits corresponding to all the priority levels (High, Mid and Low) are stored in a common buffer. Let buffSize represent all the available space in an input buffer. buffAvail indicates whether there is space available in the input buffer. The stored data flits are forwarded to the router based on their scheduling criteria. For example, in case of priority-based scheduling, High priority flits are forwarded before the Mid or Low priority flits, etc. Table 1 shows the classification of flit types.

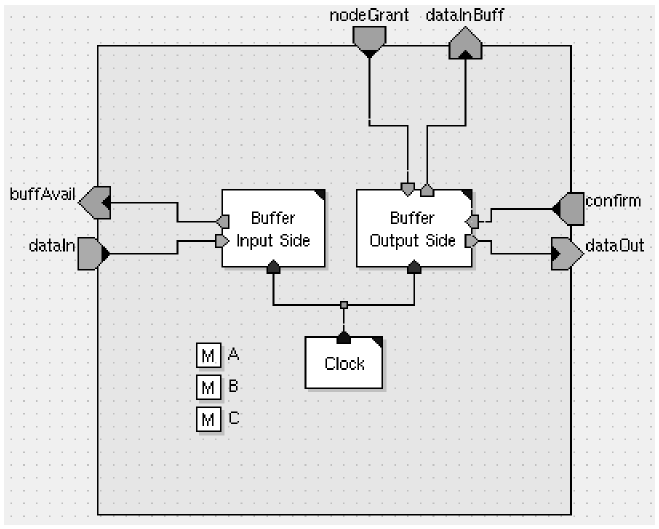
Bit combination	Flit type
00	No flit
01	High priority flit
10	Mid priority flit
11	Low priority flit

T

Table 1: Flit Priority Bit Combinations.



(a) (b)



(c)

Figure 7: MLD Implementation of the InputBuffer Class: (a) Input Side; (b) Output Side; (c) Combined Input and Output Sides.

The input buffer is virtually divided into three different buffers A, B and C (see Figures 7 (a) and (b)). We have provided flexibility in the size of these virtual buffers (the combined size is a fixed user-defined parameter). The input side (i.e. Figure 7 (a)) is responsible for checking the available buffer space and allocates memory space for an incoming flit. The output side (i.e. Figure 7 (b)) forwards a flit and releases the allocated memory.

We have implemented a handshaking protocol for forwarding a flit. The availability of data flits in the input buffer for further transmission is indicated by dataInBuff. If a grant comes in response to this request (nodeGrant), the flit stored in the buffer is forwarded to the corresponding Router. Table 2 shows the bit allocation of nodeGrant.

Bit combination	Flit type
00	No flit
01	High priority flit
10	Mid priority flit
11	Low priority flit

Table 2: nodeGrant Bit Allocation.

On receipt of nodeGrant, a data packet is forwarded by the output side of the input buffer through dataOut. Figure 7 (c) shows the complete implementation of the input side and the output side of the InputBuffer class. Three virtual buffers as shown in Figures 7 (a) and (b) are represented as memories (M) in Figure 7 (c). A flit is not removed from the input buffer until a confirmation (via confirm) is received from the scheduler (from the output side of Figure 7 (c)). If confirm is not received, the data flit is not removed from the input buffer; however, it will be queued for later forwarding. We provided three virtual channels (VC) per buffer. A VC controller inside the input buffer updates these virtual channels. We implemented the InputBuffer class with a discrete-event model of computation. The DE domain facilitates the signaling protocol and is thus a suitable choice for implementing handshaking protocols.

The customizable parameters for the InputBuffer class are the buffer size and the scheduling criteria. We can change the buffer size to any value. By changing the buffer size, we can understand its impact on latency, area and therefore the silicon cost. A buffer forwards data to the next block based on its scheduling criteria. We provided three scheduling criteria (also referred to as service levels, SL): first-come-first-serve (FCFS), priority-based (PB), and priority-based-round-robin (PBRR). In the FCFS scheduling criterion, all the data packets are treated in the same way. A data flit is forwarded to the next block based on its arrival time. The data flit with the earliest arrival time will be forwarded first. In the PB scheduling criterion, the input buffer first forwards all data packets with High priority, then those with Mid priority. It forwards Low priority data packets only when there are no High or Mid priority data packets present in the input buffer. In the PBRR scheduling criterion, the input buffer forwards data packets with different priorities in a specific rotation pattern. It first forwards a High priority packet, then a Mid priority packet, followed by a Low priority packet. This cycle is repeated throughout the simulation.

4.4 Scheduler Class

A scheduler is instantiated from the Scheduler class. The data and control parts of a node have been separated (the Router class handles the data part and the Scheduler class handles the control signals) to manage concurrency issues and make the design more scalable and reusable. The actual data flows from one node to another through a router. The path of the actual flow of this data is defined by a scheduler. A scheduler is also responsible for synchronizing the input buffer with the router and the router with the output buffer while receiving and transmitting the data further. It schedules the incoming requests for data transmission to the next node, by checking for the availability of the output data path and by arbitrating the requests from various input buffers associated with it. The Scheduler class has been implemented in the DE domain. A scheduler is mainly responsible for synchronization, and thus the DE domain is the ideal MoC for its implementation. Figure 8 shows the MLD implementation of the Scheduler class interfaced with five buffers on input and output side. The scheduler is connected to five instances of InputBuffer (one for each direction in the 2-D mesh network and a fifth buffer for the local Producer class connected through a NI) and, similarly, five instances of OutputBuffer on the output side.

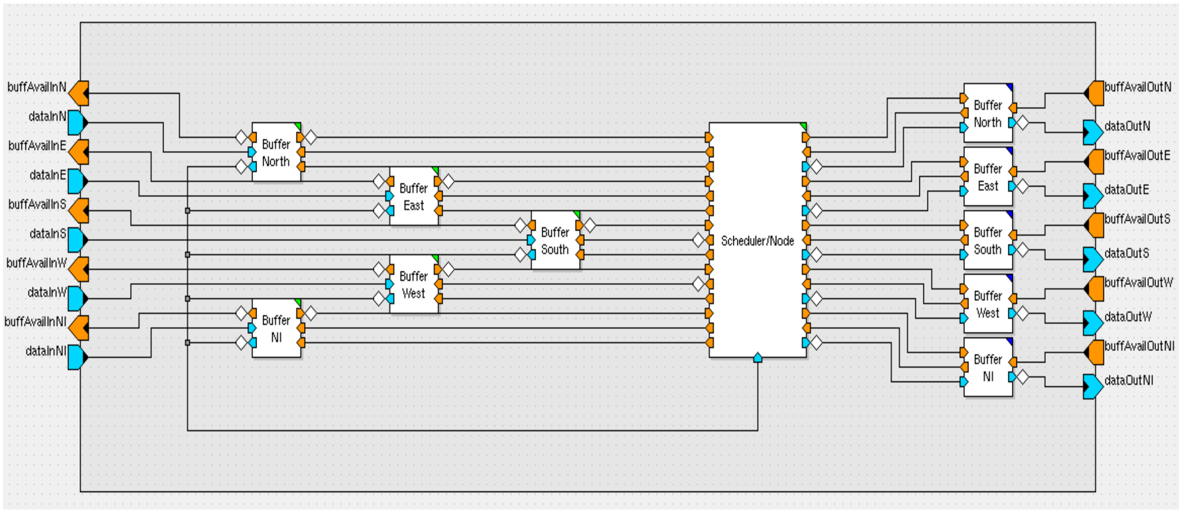


Figure 8: MLD Implementation of the Scheduler Class Interfaced with Five Input and Output Buffers.

A scheduler accepts the requests from an input buffer via dataInBuff (the input signal on the left side of Figure 8) and allocates the data path by asserting nodeGrant (the output signal on the left side of Figure 8). The data path allocation is based on the availability of an output buffer and the route. We have embedded multiple algorithms in the Scheduler class as discussed in the previous section. A scheduler will select an input buffer from multiple input buffers requesting for transmission of a data packet. The router informs the scheduler about the physical output path for flit transmission via outputPort. Availability of the data path is acknowledged by assertion of confirm. This interaction between the Scheduler and Router classes is shown in Figure 9. The

Scheduler class is implemented in two different MoC. The control part of the scheduler has been implemented with FSM. This FSM interacts with the DE domain for proper handshaking with the input buffer and router on the input side and the output buffer on the output side of the scheduler.

The customizable parameter for the Scheduler class is the scheduling criterion. The Scheduler class supports different scheduling criteria: first-come-first-served (FCFS), round-robin (RR), priority-based (PB), and priority-based-round-robin (PBRR). Thus, in a network we can have a combination of scheduling algorithms.

4.5 Router Class

A router (shown in Figure 9), instantiated from the Router class, determines the output path and handles the actual data transfer on the implemented backbone. The router receives a certain number of data flits per unit time and is constrained by the data bandwidth for transmitting a fixed number of flits at the output. Thus, the Router class has been implemented in the SDF domain. A dimension-order routing protocol was implemented in the Router class for determining the output path. We have provided customization in routing algorithms as discussed earlier. Upon receipt of data, a router extracts the destination information and determines the physical output port for transmitting the data. This output port address is sent to the corresponding scheduler, which determines the availability of this port. Upon its availability, data flits are then forwarded to the corresponding output buffer for this port.

The type of routing algorithm is a parameter for the Router class. It can be set to: X-direction-first, Y-direction-first, and XY-random. In the X-direction-first algorithm, the data is routed to the X-direction first provided there is the possibility of the data to be routed to the Y-direction as well. The Y-direction-first algorithm works similarly. In the XY-random algorithm, if there is a possibility for the data to be routed to the X-direction as well as the Y-direction, the direction is chosen in a randomized fashion with the same likelihood for the two choices.

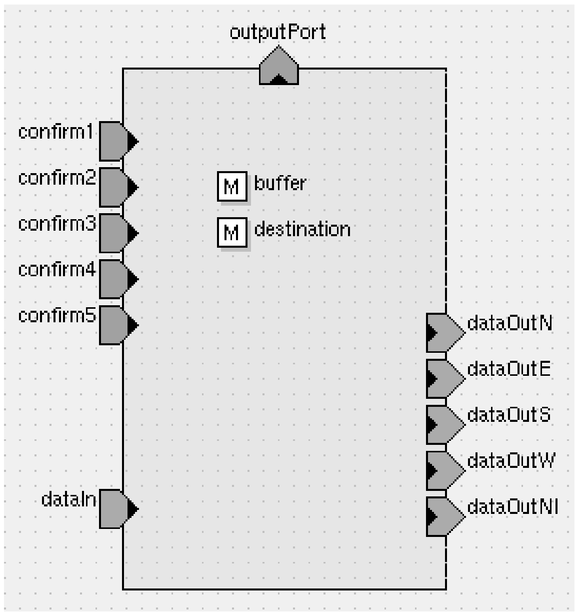


Figure 9: MLD Implementation of the Router Class.

4.6 OutputBuffer Class

An output buffer (shown in Figure 10), instantiated from the OutputBuffer class, accepts the incoming flits from the router and forwards these flits to the input buffer of the next node. It is implemented in the form of two concurrently executing state machines. The received flits are stored in the output buffer memory. The input state machine accepts and stores the data flits while there is available memory in the output buffer. Upon the request of the scheduler by reqBuffAvail, the availability of buffer space in the output buffer is signaled by buffAvail. The output state machine senses the request for transmitting data from the output buffer of the next router via outputBuffAvail of that output buffer and forwards the data flit, if that signal was asserted.

The customizable parameters for the OutputBuffer class are the same as those for the InputBuffer class.

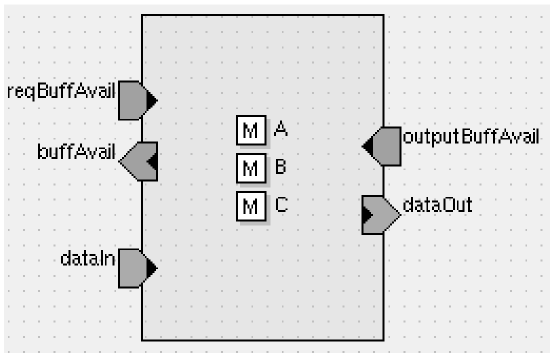


Figure 10: MLD Implementation of the OutputBuffer Class.

4.7 Consumer Class

A consumer, instantiated from the Consumer class, comprises a computing resource and a network interface (NI). A consumer accepts data flits, strips off the header information, and forwards the remainder of the data to its internal computing resources. A consumer consumes data packets. Thus, as we did for the Producer class, we have implemented the Consumer class in the SDF domain.

5. NUMERICAL RESULTS

Figure 11 shows the resulting 4x4 mesh-based NoC after system integration. We have simulated this NoC model with different packet injection rates (varying from 0.13 to 1.0), buffer sizes (1 through 10), and scheduling criteria (PB, PBRR, FCFS, RR). From these simulations we have estimated the High, Mid, and Low priority latencies. We have further implemented the NoC on a FPGA and estimated the total area and areas for each component.

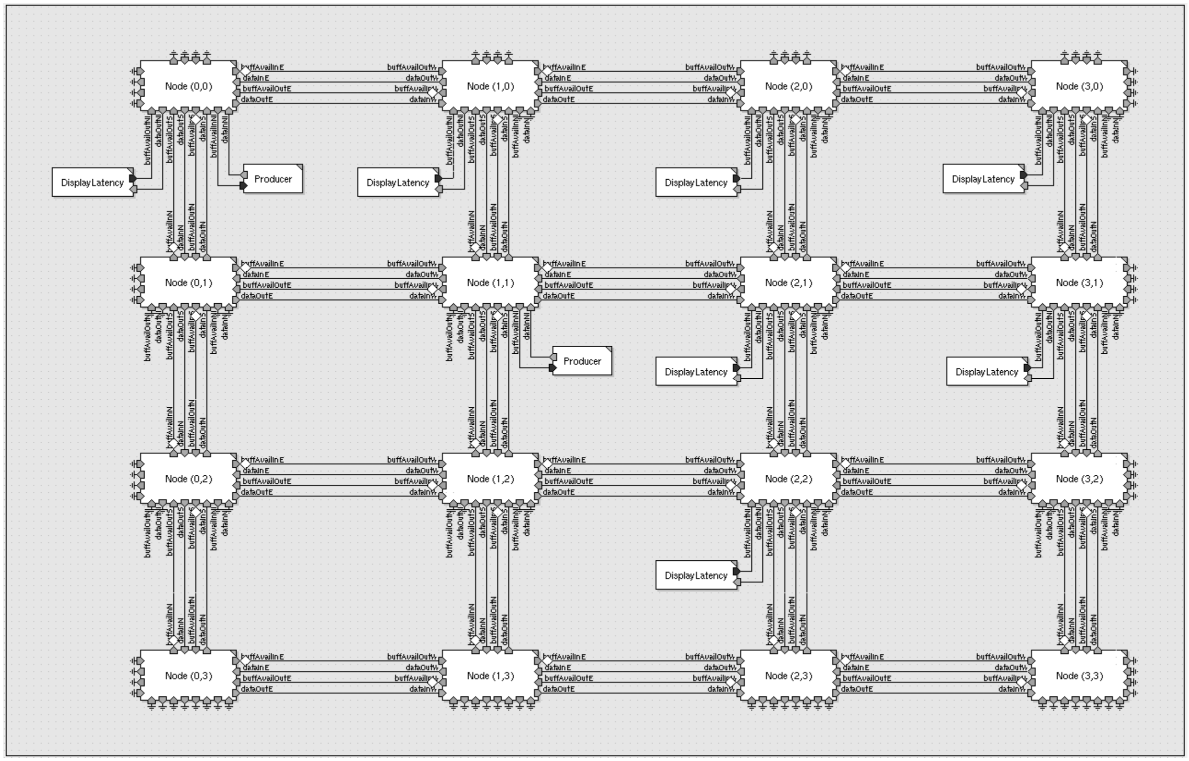


Figure 11: MLD Implementation of 4×4 Mesh NoC.

5.1 Latency Results via Simulation

The minimum latency for a single hop will be 6 clock cycles due to synchronization signals (concurrency cost) among different components: the 1st clock cycle is for storing data flits into the input buffer (buffAvail); the 2nd clock cycle is for requesting the data output to the scheduler (dataInBuff); the 3rd clock cycle is for receiving the grant signal from the scheduler (nodeGrant); the 4th clock cycle is for forwarding the data to the router (dataOut); the 5th clock cycle is for confirming the output path availability (reqBuffAvail); the 6th clock cycle is for forwarding the data packet to the output buffer (dataOut).

To simulate the NoC, we injected 10,000 flits into the network. High priority flits are used for transmitting control signals such as MemRD (Memory Read), MemWR (Memory Write), IORD (Input/Output Read), IOWR (Input/Output Write), and interrupts among others, while Mid priority flits and Low priority flits are used for transmitting real-time data and non-real time block transfers, respectively. For the network simulation, control signals were assumed to account for 10 % of the total data traffic flow, with real-time data and non-real time data accounting for 20 % and 70 %, respectively. However, this traffic load can be altered as it is a customizable parameter.

The final implementation of the NoC network was simulated for various scheduling criteria (FCFS, PB, PBRR, and RR) for varying buffer sizes (from 1 to 10) and varying packet injection rations as well. The FCFS scheduling algorithm does not prioritize the data packets. Thus, packets with all three priorities suffer almost the same latency. As the buffer size increases from 1 to 10, the data latency also increases from 14 to 107. With a larger buffer size, a data packet has to wait in the buffer for a longer time. Thus, larger buffer sizes lead to higher data latencies. In PBRR scheduling, High priority data packets must secure the lowest latency while Low priority packets have the highest latency. The data packet latencies for PBRR scheduling criteria vary from 8 to 30 clock cycles for High priority, from 15 to 36 clock cycles for Mid priority, and from 15 to 140 clock cycles for Low priority. The main advantage of the PBRR algorithm is its capability to serve all the five input buffers connected with the scheduler at the same time. Each input buffer

receives an equal scheduler response time. The latency results for PBRR scheduling are very similar to those for RR scheduling: indeed, the PBRR scheduling algorithm rotates and serves each input buffer in a RR fashion. However, it provides better results when more than three input buffers are producing data at the same time. In such a scenario, RR scheduling will not be able to deliver the High priority data in time. However, PBRR will work effectively under such a condition.
In a real-time embedded system, a system must obey real-time requirements. The NoC must deliver real-time data and control signals required for this data processing in a timely manner. Consequently, we must select scheduling criteria that can perform this task. Figure 12 shows High priority data latency results against buffer sizes for different scheduling criteria.

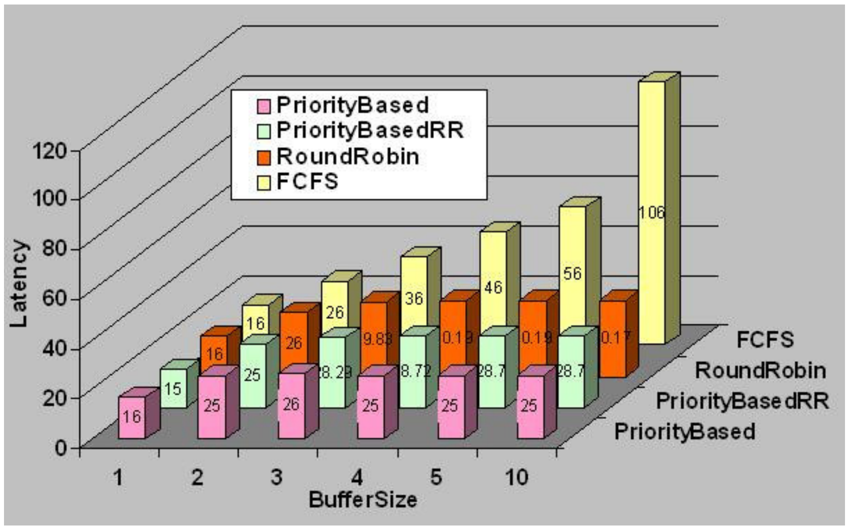


Figure 12: High priority data latency vs buffer size for different scheduling criteria.

It can be seen that the FCFS and RR scheduling algorithms may not always deliver High priority data packets in real-time bounds. Thus, we should use the PBRR or PB scheduling criteria for this NoC architecture. We used Low priority data latency results to finally choose between PBRR and PB scheduling. Figure 13 shows Low priority data latency against buffer sizes for PBRR and PB scheduling.

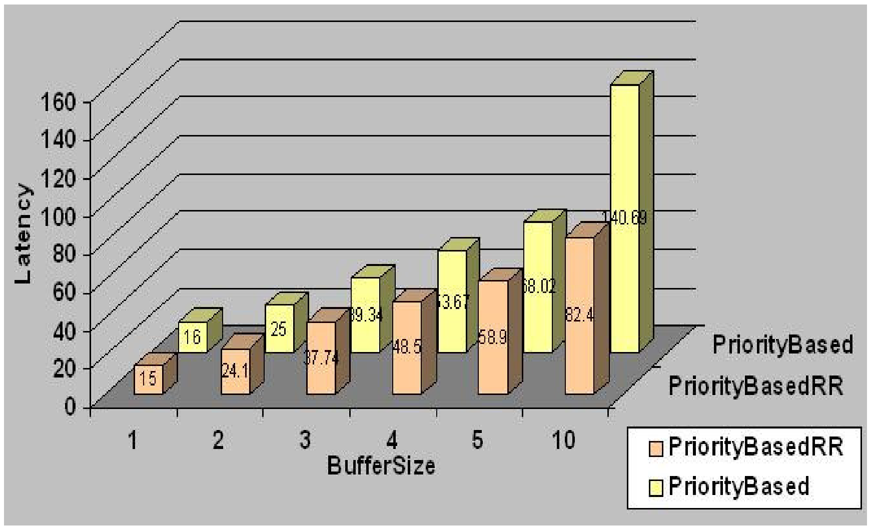


Figure 13: Low Priority Data Latency vs Buffer Size for Different Scheduling Criteria.

PB scheduling delivers the High priority data and Mid priority data in a timely manner. However, it has a higher latency for low priority data. Thus, it is best to use PBRR as the scheduling criterion for this NoC architecture.

5.2 Area Results via FPGA Implementation

We implemented all the Network-on-Chip (NoC) classes with Field Programmable Gate Arrays (FPGA) to extract the total NoC area and area information of each component. Table 3 shows the area result for a 32-bit and 64-bit Input Buffer and Output Buffer implementations. These area results are not directly proportional to the buffer size i.e. a linear relationship between buffer size and number of gates does not exist. This is mainly because of two reasons:

1. Our Input Buffers and Output Buffers have scheduling capability, along with the functions of storing and forwarding the data packets, as discussed in chapter 5. This scheduling circuit does not depend upon the buffer size. Therefore, buffer of a size two will not have double the number of gates as compared to a buffer of size one.

2. The internal architecture of an FPGA is divided into several Configurable Logic Blocks (CLBs) along with other components such as Digital Clock Manager (DCM) and block Random Access Memory (BRAM). A CLB is further divided into four slices. Architecture of each CLB is identical. The number of gates is calculated by multiplying the number of occupied slices with the total number of gates in a single slice. Thus, the equivalent FPGA implementation will count the total number of gates for a slice even if it is not fully occupied. Similarly, if we use a part of BRAM for storing the data packet/flits, then we account for total number of gates for that BRAM.

Consequently, we will not see a linear relationship between the buffer size and the number of gates.

Buffer Size	**No. of Gates for 32-Bit Buffer**	**Number of Gates for 64-Bit Buffer**
1	7,863	10,686
2	9,162	11,581
3	9,622	12,423
4	9,812	12,310
5	9,924	12,510
10	10,680	13,273

Table 3: Total Number of Gates for Different Buffer Sizes

Table 4 shows area result for different scheduling criteria (Round Robin (RR), Priority Based (PB), First Come First Serve (FCFS), and Priority Based Round Robin (PBRR)) and Router. Router handles the actual data transfer, thus has dedicated lines for sending and receiving data bits. Thus, Router takes more number of gates than Scheduler.

Component	Number of Gates
PBRR Scheduler	5,954
FCFS Scheduler	3,155
RR Scheduler	3,674
PB Scheduler	3,554
Router	9,589

Table 4: Total Number of Gates for Different Scheduling Algorithms in Scheduler and Router.

5.2.1 Area Results via FPGA Implementation: Impact of Buffer Size on NoC Area: There are eighty input buffers and eighty output buffers in a 4×4 mesh based NoC. Thus, buffer size is a key parameter for deciding the number of gates used in a NoC architecture. Figure 14 shows the High, Mid and Low data latencies against different 64-bit buffer sizes for PBRR scheduling.

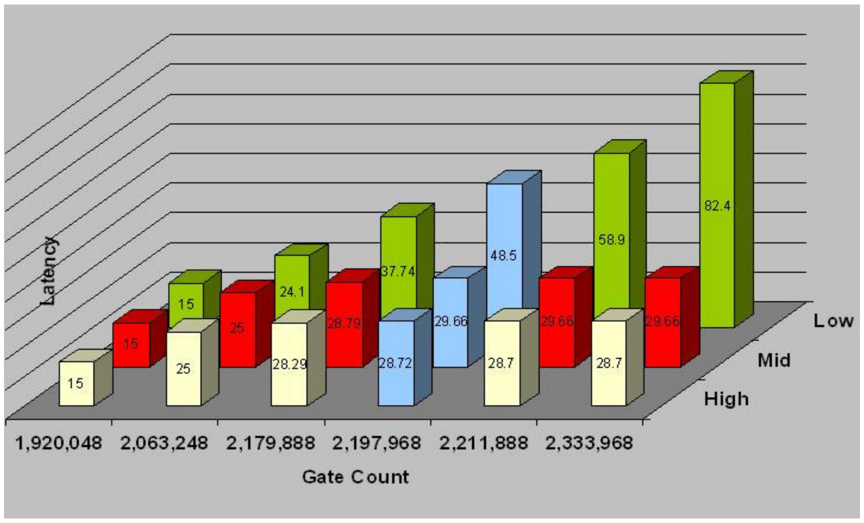


Figure 14: Latency vs NoC area with PBRR scheduling and different buffer sizes.

We also provided the total NoC area (gate count) for different buffer sizes with PBRR scheduling criteria. A buffer size of 10 produces a high value of Low priority data latency (82.4 clock cycles). Thus, we do not recommend using a buffer size of 10. Buffer sizes of 1 and 2 will have lower values of High priority data latency, however they may not provide sufficient amount of buffering needed in a network. The High and Mid priority data latencies for buffer sizes of 3, 4, and 5 are almost similar. A buffer size of more that 5 will have a higher number of gates; consequently the leakage current will also increase. Therefore, for this NoC it is recommended to use buffer sizes of 3, 4, or 5.

5.3 Leakage Power Consumption Results for NoC Parameters

FPGAs have three main factors affecting the power consumption. These are leakage current, dynamic power, and inrush current. Leakage current is a small amount of current that exists even when the gate is idle. Such power consumption is referred as leakage power consumption. Its value depends upon the technology. In 90 nm technology this leakage current is the dominant factor. As the process gets smaller (due to fabrication technology advances), the core voltage decreases. Therefore, even if the frequency increases, the rate of increase of the dynamic power drops. However, static power has grown exponentially with the introduction of new process technologies. Thus, leakage power that was ignored earlier needs to be accounted in total power consumption.

Inrush current is the result of the internal circuitry contention when the FPGAs are powered up. These spikes can measure in multiple amperes, thus causing huge amount power consumption. However, with the advancement of FPGA technology and introduction of 90 nm FPGA technology this issue has been resolved. Further with recent advances, parts of FPGAs can be put to sleep state consuming a minimum amount of leakage power.

FPGA architecture mainly consists of CLBs, IOB (Input/Output Blocks), carry-chain adder, multipliers, BRAMs, and DCM among others. In this dissertation, we provide some insight on leakage power consumption for 1.2 V, SRAM (Static Random Access Memory) FPGAs built in 90 nm technology. The embedded version of this FPGA is also available and consists of only CLBs. We detail our leakage power consumption on this embedded version of FPGA. As explained earlier that the FPGA architecture consists of CLBs and is symmetrical in nature. So we focus our study on single CLB.

Leakage current may change by a factor of three based on the input to a circuit. For example, a 2:1 multiplexer has three inputs (2 inputs and 1 select line). Its leakage current value ranges from

2.88 mW (when all the inputs are zeros) to 0.91 mW (when all the inputs are ones). However, the total value of leakage current will also depend upon the utilization factor. Unutilized logic will consume less leakage power as compared to utilized logic. Usually, we attain a high utilization factor for CLBs and a low utilization factor for IOB. At 25o C, an FPGA consumes 4.2 µW per CLB. Table 5 lists the number of slices used for designing NoC components and the total leakage power consumption for NoC implementation with different buffer sizes and scheduling algorithms. This leakage power will entirely depend upon the FPGA technology.
.

NoC Components	Number of Slices	Leakage Power (W)
Buffer Size 1	481	0.0020202
Buffer Size 2	546	0.0022932
Buffer Size 3	594	0.0024948
Buffer Size 4	601	0.0025242
Buffer Size 5	612	0.0025704
Buffer Size 10	671	0.0028182
Router	778	0.0032676
Scheduler (PB)	290	0.001218
Scheduler (RR)	274	0.0011508
Scheduler (PBRR)	356	0.0014952
Scheduler (FCFS)	201	0.0008442

Table 5: Leakage Power Consumption for NoC Components

6. CONCLUSION

We have developed a methodology to realistically model and optimally design the communication backbone of a NoC. The model is built with reusable and customizable building blocks, which are abstract enough to facilitate rapid analysis. We used MLD simulation environment because of its support for multiple MoC, which helps with both fast model building and simulation.

Acknowledgement

This work reported in this paper was carried out as part of the “One-Pass-to-Production” research project, funded by the iDEN division of Motorola, Plantation, FL.

7. REFERENCES

1. A. Jantsch, H. Tenhunen, *“Network on Chips”* Kluwer Academic Publishers, Boston, (2003)

2. G. Desoli and E. Filippi, “*An outlook on the evolution of mobile terminals: from monolithic to modular multi-radio, multi-application platforms”*, IEEE Circuits and Systems Mag., 6(2): 17-29, 2006.
3. W. C. Rhines, *“Sociology of design and EDA”*, IEEE Trans. on Design and Test, 23(4): 304-310, 2006.

4. E. A. Lee and Y. Xiong, *“System level types for component-based design”*, Workshop on Embedded Software, California, 2001.

5. Y. Xiong and E. A. Lee, *“An extensible type system for component-based design”,* International Conf. on Tools and Algorithms for the Construction and Analysis of Systems, Berlin, Germany, 2000.

6. D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli, *“NoC synthesis flow for customized domain specific multiprocessor SoC”*, IEEE Trans. on Parallel and Distributed Systems, 16(2): 113-129, 2005.

7. S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, *“A Network on Chip architecture and design methodology”*, IEEE Symposium on VLSI, 117-124, 2002.

8. A. Agarwal and R. Shankar, *“Modeling concurrency on NoC architecture with symbolic language: FSP”*, IEEE International Conf. on Symbolic Methods and Applications to Circuit Design, 2006.

9. J. Burch, R. Passerone, and A. L. Sandivanni-Vincentelli, *“Overcoming heterophobia: modeling concurrency in heterogeneous systems”*, IEEE International Conf. on Applications of Concurrency to System Design, 13-32, 2001.

10. E. A. Lee and A. Sangiovanni-Vincentelli, *“Comparing models of computation”*, IEEE/ACM International Conference on Computer-Aided Design, 234-241, 1996.

11. A. Jantsch and I. Sander, *“Models of computation and languages for embedded system design”*, IEEE Proceedings on Computers and Digital Techniques, 114-129, 2005.

12. A. Girault, B. Lee; E.A. Lee, *“Hierarchical finite state machines with multiple concurrency models”*, IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems, 18(6): 742-760, 1999.

13. A. Agarwal and R. Shankar, “A Layered Architecture for NoC Design methodology”, IASTED International Conf. on Parallel and Distributed Computing and Systems, pp. 659-666, 2005.

14. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, *“Performance evaluation and design trade-offs for network-on-chip interconnect architectures”*, IEEE Transaction on Computers, 54(8):1025-1040, 2005.

15. A. Agarwal, R. Shankar, C. Iskander, G. Hamza-Lup, *“System Level Modeling Environment: MLdesigner”*, 2nd Annual IEEE Systems Conference, Montreal, Canada, 2008.

16. Bertozzi and L. Benini, *“Xpipes: A network-on-chip architecture for gigascale systems-on-chip”*, IEEE Circuits and Systems Magazine, 4(1):18-31, 2004.

17. S. J. Lee, K. Lee, S. J. Song, and H. J. Yoo, *“Packet-switched on-chip interconnection network for system-on-chip applications”*, IEEE Transaction on Circuits and Systems II, 52(6), :308-312, 2005.
18. W. J. Dally and B. Towles, *“Route packets, not wires: on-chip interconnection networks”*, IEEE International Conference on Design and Automation, 2001.
19. J. Burch, R. Passerone, A.L. Sandivanni-Vincentelli, “Overcoming heterophobia: modeling concurrency in heterogeneous systems”, IEEE International Conference on Application of Concurrency to System Design, pp. 13-32, 2001

20. G.H. Hilderink, *“Graphical modeling language for specifying concurrency based on CSP”*, IEEE Proceedings on Software Engineering, 150(2): 108 – 120, 2003.

21. S. Chrobot, *“Modeling communication in distributed systems”*, IEEE International Proceeding in Parallel Computing in Electrical Engineering, 2002

22. T. Murphy, K. Crary, R. Harper, F. Pfenning, *“A symmetric modal lambda calculus for distributed computing*”, 19th Annual IEEE Symposium on Logic in Computer Science, 2004.

23. A. Girault, B. Lee; E.A. Lee, *“Hierarchical finite state machines with multiple concurrency models*”, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, 18(6): 742-760, 1999.

24. M. Barrio, P. De La Fuente, *"IEEE International Computer Science Conference on Software Engineering*", 1997.

25. J. Magee, J. Kramer, *"Concurrency State Models and Java Programs*", West Sussex England, John Wiley & Sons, (1999).

Intention-based Ranking for Surface Realization in Dialogue Systems

Aida Mustapha aida@fsktm.upm.edu.my
Faculty of Computer Science and
Information Technology
University Putra Malaysia
43400 UPM Serdang, Selangor, Malaysia

Md. Nasir Sulaiman nasir@fsktm.upm.edu.my
Faculty of Computer Science and
Information Technology
University Putra Malaysia
43400 UPM Serdang, Selangor, Malaysia

Ramlan Mahmod ramlan@fsktm.upm.edu.my
Faculty of Computer Science and
Information Technology
University Putra Malaysia
43400 UPM Serdang, Selangor, Malaysia

Mohd. Hasan Selamat hasan@fsktm.upm.edu.my
Faculty of Computer Science and
Information Technology
University Putra Malaysia
43400 UPM Serdang, Selangor, Malaysia

Abstract

A new intention-based ranking is proposed to cater for intentionality in ranking dialogue utterances, as opposed to surface-based ranking using surface linguistic features in utterances. This is because utterances may be in the form of a sentence, a phrase, or just a word; hence basis for ranking must be on assessment of intentions, regardless of length of utterance and grammar rules. Intention-based ranking model is tested and compared with surface-based models on 15 response classes in theater domain. The results from comparative experiments show consistent accuracy distribution in intention-based ranking across all response classes with average of 91%. On the contrary, ranking accuracy for surface-based ranking is not uniform across the response classes, showing the dependency on surface representation of utterances in individual response class.

Keywords: Overgeneration-and-ranking, Ranking, Classification, Intention, Dialogue systems.

1. INTRODUCTION

Statistical approaches to surface realization sidestep the linguistic decision-making process by applying statistical learning in the surface generator itself, as opposed to the deterministic knowledge-based approach. This approach is known as overgeneration-and-ranking [1], which relies on corpus to furnish semantically related sentences through surface linguistic features of sentences. The principle objective is to help reducing the amount of syntactic knowledge to be hand-coded manually as required by knowledge-based approach. The effort required to construct grammar for the overgenerator is also very minimal; enough for it to generate lattices. Due to the minimal generation technology, an additional task of ranking is necessary. The need for ranking arises to discriminate out candidate sentences that are ungrammatical, unintelligible or at least not fluent by means of language models.

Langkilde and Knight [1] and Langkilde [2, 3] focused on learning surface structure of sentences at the syntactic level, while subsequent researches [4, 5] extended learning into semantic level through incorporation of semantic information. This is essentially a mapping from semantic to syntactic. For instance, Bangalore and Rambow [4] use dependency tree labeled with extended synonyms rather than lexemes, while Varges [5] utilizes semantic mark-up in constructing its grammar base. Similarly, Ratnaparkhi [6] and Oh and Rudnicky [7] apply language models on ranking generation templates. Nonetheless, the motivation remains, which is to learn and regenerate the sentences based on surface linguistic features.

The main limitation of overgeneration-and-ranking is that, it is computationally expensive to overgenerate in setting up the band of realization candidates, either through simple grammar-rules or statistical means like *n*-grams [5]. While knowledge-based approach through grammar is not usually fast enough for use in dialogue systems [8], overgeneration is also not necessary for generation of dialogue utterances due to two main reasons. Firstly, dialogue utterances are typically short, single-sentenced, and are often incomplete. They can take form of a sentence, a phrase, or just a word. Secondly, dialogue utterance bears individual intention. Even if the surface form is grammatically incorrect, an utterance fares well as long as it satisfies the intentions of the utterance it is responding to.

Language models, although robust, also have built-in bias to produce short strings because the likelihood of a string of words is determined by the joint probability of the words [9]. This is clearly not desirable for generation of dialogue utterances because all utterance should be treated based on assessment of the intentions, regardless of length, in fact, regardless of grammar. While the output may be inarguably sophisticated, the impact may be not as forceful. We believe that ranking dialogue utterances requires more than statistical distributions of language, but more intuitive in the sense that ranking model incorporates intentionality to maintain coherence and relevance, regardless of the surface presentation.

Intention-based ranking [10] is taking pragmatic approach to assessing dialogue utterances. Different from previous ranking models that deal with language models and semantic features, intention-based ranking focuses on finding the best utterance based on the semantic and pragmatic knowledge they represent. The knowledge exists in the form of (1) semantics from user utterances and (2) intentions, semantics, and domain informativeness from response utterances. The utterance with highest probability score is said “relevant” with respect to input utterance when topic of response utterance satisfies the intention of user utterance.

The remainder of this paper is organized as follows: Section 2 will present four different ranking models; three of the models are surface-based while the last is the proposed intention-based ranking model. Section 3 will provide experimental background by introducing the corpus and dataset used during the experiment. Finally, result findings are reported and discussed in Section 4 before the paper is concluded in Section 5.

2. RANKING MODELS

Surface-based ranking under the overgeneration-and-ranking methodology involves a task to rank all sentences or utterances (called lattices) resulted from an overgeneration process that capitalizes on semantic and surface linguistic features obtained from the corpus. The goal is to find the highest probability utterance ranked as output of the process. Similarly, the goal of intention-based ranking is also to find an utterance with the highest probability as the output. Nonetheless, while surface-based ranking may consider hundreds or thousands of lattices at one time, intention-based ranking only consider utterances in specific, individual response class, resulted from the classification process under the classification-and-ranking methodology.

This section presents decision rules for all surface-based ranking models that we consider; which are *n*-grams language model, maximum entropy with language model, and instance-based learning model. At the end of the section is the decision rule for the proposed intention-based ranking model that capitalizes on intentions rather than surface features.

2.1 *N*-grams Language Model

A language model is a statistical model of sequence of words, whereby probability of a word is predicted using the previous *n*-1 words. Following *n*-gram ranking [1, 11, 12], response utterances are trained by a trigram model through counting and normalizing words inside the utterances. Consider the following response utterance:

r = "*Yes there are still 826 seats available.*"

A trigram generation model for response utterance *r* will record the pair (still, 826) or (826, seats) and the triple (still, 826, seats) or (826, seats, available). The model will then estimate the probability for *P(r)*, which is the estimated probability for response *r* based upon some count *C*. Therefore, to estimate the probability that "seats" appears after "826", the model divides the count of the pair (826, seats) by the triple (826, seats, available). This ratio is known as relative frequency or maximum likelihood estimation.

Equation 1 shows estimation of probabilities based on relative frequency of words inside response utterance *r* and the final probability of each response utterance, where *n* is total number of running words in the utterance and w_1^n is *n*-gram of $w_1 \ldots w_n$ instances in training set.

$$P(w_i \mid w_{i-2}, w_{i-1}) = \frac{C(w_{i-2} w_i)}{C(w_{i-2} w_{i-1})} \tag{1}$$

$$P(w_1^n) = \prod_{i=1}^{n} P(w_i \mid w_{i-2}, w_{i-1})$$

Based on the equation, response utterances are ranked using the negative log probabilities with respect to the language model. Back-off smoothing was applied for unobserved *n*-grams (i.e., *n*-grams that do not exist in training set), which is bigram in case of zero-probability trigram. In case of our ranking experiment, we employed features extracted by a trigram language model.

2.2 Maximum Entropy with Language Model

Similar to language models of *n*-gram, implementation of Maximum Entropy (ME) ranking [6, 12] is also surface-based, which means they rely on surface features like frequencies of *n*-grams. Nonetheless, because the ME model is trained on a corpus of existing generation templates, this provides semantic knowledge to ranking as captured by the template attributes. The basic assumption of this model is that, the best choice to express any given meaning representation (in the form of attribute-value pairs) is the word sequence with highest probability that mentions all the input attributes exactly once [6].

Feature function $f(w_i, w_{i-1}, w_{i-2}, attr_i)$ is constructed based on local information captured by *n*-grams and non-local information represented by the input attributes. The *n*-gram, which in this model is a bigram, enables the system to learn the word choice (lexical choice) and word order (attribute ordering) directly from the corpus. The ME probability model with bigram and domain attributes are shown in Equation 2.

$$p(w_i \mid w_{i-1}, w_{i-2}, attr_i) = \frac{\prod_{j=1}^{k} \alpha_j^{f(w_i, w_{i-1}, w_{i-2}, attr_i)}}{Z(w_i, w_{i-1}, w_{i-2}, attr_i)} \tag{2}$$

where $Z(w_i, w_{i-1}, w_{i-2}, attr_i) = \sum_{w'} \prod_{j=1}^{k} \alpha_j^{f(w_i, w_{i-1}, w_{i-2}, attr_i)}$

Based on the equation, ranking is performed using probability of the sequence of words $W = w_1 \ldots w_n$ given a set of attributes *A* and length of utterance *n* in the following Equation 3. To perform our ranking experiment, we abstracted out response utterances into response templates as the set of domain attributes *A*.

$$P(W \mid n, A) = \prod_{i=1}^{n} p(w_i \mid w_{i-1}, w_{i-2}, attr_i) \tag{3}$$

2.3 Instance-based Learning

Instance-based approaches are lazy, supervised learning methods that simply store the training set examples (instances) and use them directly when a new input is to be processed. At run time, the new inputs are compared to each instance in the training set (instance base). An instance-based ranker [5] scores the candidates according to their similarity to instances in the instance based taken from the training corpus. Varges [5] uses standard information retrieval techniques for representation of instances, which is *tf.idf*. The equation for *tf.idf* is represented by Equation 4, whereby $f_{i,j}$ is the term frequencies (*tf*) and $\log_2 N/n_i$ is the inverse document frequency (*idf*).

$$w_{i,j} = tf \cdot idf = f_{i,j} \times \log_2 \frac{N}{n_i} \tag{4}$$

where $f_{i,j} = f(w_i, w_{i-1}, \ldots, w_{i-n})$

For the case of our ranking experiment, *n* is the number of words in a response utterance, while *N* is the total number of response utterances in a particular class. Therefore, term frequency (*tf*) refers to individual word frequency in a particular utterance and inverse document frequency (*idf*) refers to inverse utterance frequency in a collection of response utterance *N*. After we represented all user utterances in the form of weights $w_{i,j}$, we used a simple distance measure (normalized Euclidean distance) to find the training instance closest to the given test instance, and predicts the same class as this training instance, following nearest-neighbor approach. If multiple instances include the same (smallest) distance to the test instance, the first one found is used.

Similar to the first two approaches, *n*-grams and ME augmented with *n*-grams, instance-based ranking is also surface-based, which the goal is to find the best sequence of words that forms the templates with semantic annotation tags (attributes) in place rather than the actual values for the attributes. The ranking treats the attributes just like any other words in the templates because all utterances are represented in the form of weights $w_{i,j}$.

2.4 The Proposed Intention-based Ranking

Ranking is performed on response utterances $\{r_1 r_2 ... r_R\}$ from the set of response *R*. All the response utterances are classified together based on topical contributions of each individual utterance. The goal of ranking is to output a single response utterance $r \in \{r_1 r_2 ... r_R\}$ in respond to the user; by choosing a response with the highest probability score. The probability model is defined over *R* x *S*, where *R* is the set of possible response utterances $\{r_1 r_2 ... r_R\}$ and *S* is the set of corresponding features to each response utterances.

The set *S* consists of both local and global knowledge for each utterance in the response database *R*. Local knowledge are features extracted from response utterances in training corpus, which includes intentions (speech acts and grounding acts), semantics (topic and focus of response utterance), and domain informativeness (domain attributes i.e. *title*, *genre*, or *date*). Global knowledge is supplied by focus of attention in user utterances. Local and global variables used in intention-based ranking model are described in Table 1.

No	Feature	Desciptions
1	*rTopic*	Topic of conversation in response
2	*rFocus*	Focus of attention in response
3	*rFlf*	Speech act for response
4	*rBlf*	Grounding act for response
5	*rDa*	Domain attributes in response
6	*uFocus*	Focus of attention in user utterance

TABLE 1: Local and Global Knowledge for *R.*

Using both local and global features to model the probability distribution, each response utterance in the training data is defined in the form of *M* feature functions $f_m(r,\{r_1 r_2 ... r_R\}, s)$ where $r \in R$, $s \in S$ and $m = 1,\ldots,M$. The probability model of response utterance *r* is conditioned to features *s*, where λ_m are the weights associated with each feature *m* where *Z(s)* is the normalizing function as shown in Equation 6.

the normalizing function $Z(s)$ is defined in Equation

$$p(r \mid \{r_1 r_2 ... r_R\}, s) = \frac{1}{Z(s)} \exp\left[\sum_{m=1}^{M} \lambda_m f_m(r, \{r_1 r_2 ... r_R\}, s)\right] \tag{6}$$

where $$Z(s) = \sum_{r'} \exp\left[\sum_{m=1}^{M} \lambda_m f_m(r', \{r_1 r_2 ... r_R\}, s)\right]$$

Given the modeling equation, we arrive at the decision rule as shown in Equation 7.

$$\hat{r} = \arg\max_{r \in R} \left[p(r \mid \{r_1 r_2 ... r_R\}, s)\right] \tag{7}$$

3. EXPERIMENTS

The objective of this paper is to test and compare four different statistical ranking models: an *n*-gram language model, maximum entropy (ME) augmented with language model, an instance-based learning model, and the proposed intention-based model. The corpus used in the experiment is called SCHISMA, an acronym derived from the Dutch *SCHouwburg Informatie Systeem*, a theater information and ticket reservation system [13]. Figure 1 shows an extract of SCHISMA dialogues.

U:	What will be on in the theater next week (19 March)?	[1]
S:	There is no show on that date.	[2]
U:	And on 18 March?	[3]
S:	In the period 18 March 1994 until 20 March 1994 you can go to Deelder Denkt and Indonesian Tales.	[4]
U:	At what time does Deelder start?	[5]
S:	The show starts at 20:00.	[6]
U:	How much does it cost	[7]
U:	and are there still places?	[8]
S:	Do you have a reduction card?	[9]
U:	No	[10]
S:	The price for the show "Deelder Denkt" is f26,00.	[11]
S:	And there are still 82 places free.	[12]

FIGURE 1: SCHISMA Dialogue Extract.

SCHISMA is constituted by 64 text-based dialogues of varied length. In total, there are 2,047 individual utterances in 1,723 turns. 920 utterances are user contributions and 1,127 utterances are system contributions. 920 response utterances are classified into 15 response classes based on topical contributions of the corresponding response utterances by the system [10].The list of response classes is shown in Table 2.

NO	RESPONSE CLASS	NO. OF INSTANCES
1	Title	104
2	Genre	28
3	Artist	42
4	Time	32
5	Date	90
6	Review	56
7	Person	30
8	Reserve	150
9	Ticket	81
10	Cost	53
11	Avail	14
12	Reduc	73
13	Seat	94
14	Theater	12
15	Other	61
		920

TABLE 2: Distribution for Response Classes in SCHISMA Corpus.

Ranking is performed separately on response utterances (instances) in each response class as shown in Table 2. Our evaluation metric is based on recognition accuracy of the highest-ranked response utterance as compared to the dialogue corpus. In testing all surface-based and intention-based ranking models, we used the same training and testing dataset of response classes in SCHISMA. In other words, the accuracy of ranking is evaluated by checking if the response utterance returned as the top-ranked response is correct or otherwise, with respect to response utterance in the test set.

4. RESULTS AND DISCUSSION

The performance of intention-based response generation is compared with other surface-based ranking approaches that share similar spirit of overgeneration-and-ranking.

4.1 Surface-based Ranking

This section evaluates and compares the relative performance of surface-based ranking models on 15 response classes in SCHISMA corpus. Figure 2 illustrates comparison of accuracy distribution among all techniques, which are language model (LM), maximum entropy augmented with language model ME (LM), and instance-based learning (IBL). Judging from the jagged graph curve, we can see that accuracy percentage is uneven across all response classes. This is not due to the varying number of instances in each response class, but rather due to the variation of surface structure of utterances, in the sense that how unique one utterance as compared to the other.

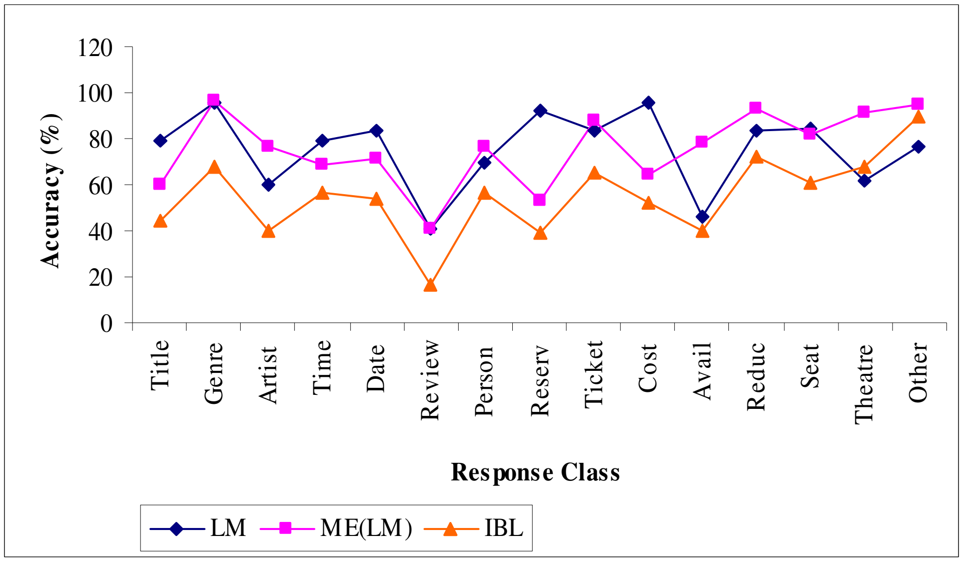


FIGURE 2: Accuracy Percentages for Surface-based Ranking.

The influence of surface forms of the utterances can be analyzed separately for all three ranking models. For trigram LM, the highest accuracy of 96% resulted from response class *genre* and the lowest accuracy of 41% from response class *review.* To study the gap between the accuracy percentages, we provide extract of instances for both response classes *genre* and *review* in Figure 3 and Figure 4 respectively.

[1] Shows fall in the following genres: Ballet, Cabaret, Dancing, family show, Musical, Music, Opera, Dramae and narrator lunch.
[2] The shows fall into the following genres: Ballet, Cabaret, Cabaret / Kleinkunst, Dance, Family Show, Youth-Theater, Musical, Music, Opera, Theater, Drama, Story-Telling and Reading. In which genre you interested?
[3] The shows fall into the following genres: Ballet, Cabaret, Dance, Family Show, Youth-Theater, Musical, Music, Opera, Theater, Drama, Story-Telling and Reading. In which genre are you interested?

FIGURE 3: Response Utterances in Response Class *genre*

[1] Show “Candide” falls in the genre musical. The famous (and traditional) love story of Candide is …
[2] Last year Han Romer and Titus Tiel Groenestege astonished the theatre world with 'Ockhams Scheermes', a show which tasted of “more”. The plans for “De Olifantsdracht” are still vague...

[3] "Der Vetter aus Dingsda" has rightfully provided Kunneke with international fame. The song "Ich bin nur ein armer Wandergesell" was and is being sung in many languages, though...

FIGURE 4: Response Utterances in Response Class *review*

Based on Figure 3 and 4, we can see that utterances in response class *genre* are short and more homogeneous in terms of the surface structures. Utterances in *review*, however, are lengthy and highly distinctive from one another. The extreme structure of response utterances in both classes shows how influential surface structures are to *n*-gram ranking accuracy.

Accordingly, the distribution accuracy for ME (LM) model is consistent with LM, except for response class *reserve*, *cost*, and *theater* that merit further explanation. Recall that for this model, *n*-gram information is in the form of domain attributes rather than the individual words in the response utterance. Ranking accuracy for both *reserve* and *cost* degrades as compared to LM because when two utterances carry the same number of domain attributes, this model is not able assign probability of the utterance any higher from the other. This can be seen from the following response utterance r_1 and r_2 that is abstracted out into *r'*.

r_1 = "You have reserved "Dat heeft zo'n jongen toch niet nodig", played by Herman Finkers on Sunday 28 May 1995. Commencement of show is 20:00. You are requested to collect these tickets minimum half an hour before commencement of the show."

r_2 = "You have reserved "De Olifantsdracht", played by Han Romer and Titus Tiel Groenestege on Sunday 22 January 1995. Commencement of show is 20:00. You are requested to collect these tickets minimum half an hour before commencement of the show."

r' = "You have reserved <title>, played by <artist> on <date>. Commencement of show is <time>. You are requested to collect these tickets minimum half an hour before commencement of the show."

Clearly, the count for domain attributes in *r'* does not help to discriminate r_1 and r_2 even though both utterances carry different semantic meaning altogether. This observation, nonetheless, does not affect response class *theater* even though the response utterances generally bear the same count of domain attributes. This is because utterances in this class carry the same semantic meaning, which is the address of the theater. Figure 5 shows excerpts of utterances in response class *theater*.

[1] Name: Theater Twent, Address: Langestraat 49, Post Code: 7511 HB, Place: Enschede, Tel.: 053-858500, Information: 053-858500.
[2] The address of this theater is: Name: Theater Twent, Address: Langestraat 49, Post Code: 7511 HB, Place: Enschede, Tel.: 053-858500, Information : 053-858500.

FIGURE 5: Response Utterances in Response Class *theater*

As for IBL ranking model, the distribution accuracy remains consistent with other models albeit the low accuracy performance as compared to LM and ME (LM). Because IBL [5] was originally implemented under the framework of text generation, this approach suffers the most due to *tf.idf* formalism used in representing weights in each response utterance. When features are represented as *tf.idf* weights, the discriminatory power for one response utterance against another degrades as *idf* assign highest weights to words that occur in few instances but still the weights will be weighed down by the *tf*.

4.2 Intention-based Ranking

The average accuracy for ranking across all response classes for SCHISMA corpus is 91.2%. To show the steady influence of intentions to ranking response utterance, the distribution of ranking accuracies is illustrated in Figure 6. Note that despite the uneven size of instances (response utterances) in every response class as shown previously in Table 2, the accuracy for all the classes is consistent from one another except for two classes, *reserve* and *other*, which merit further explanation.

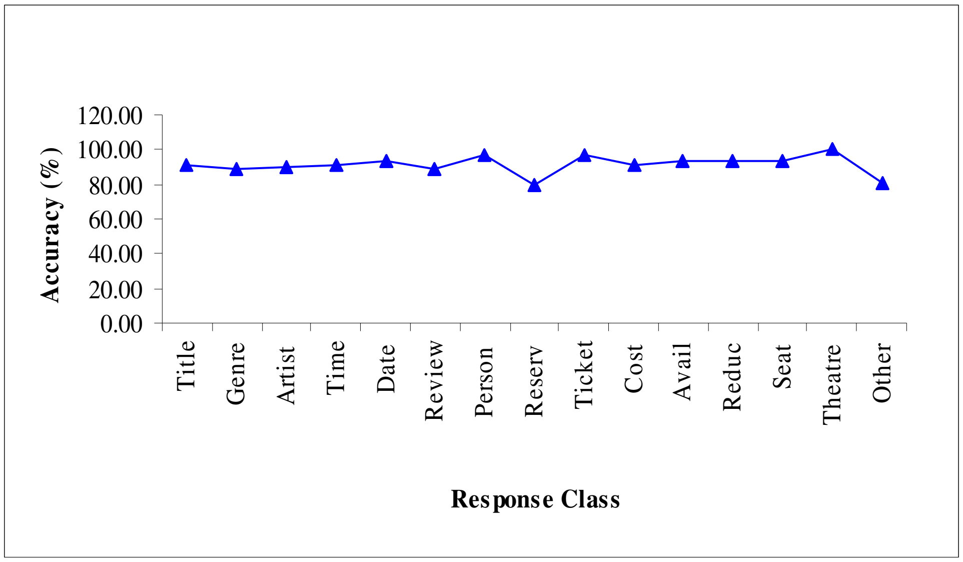


FIGURE 6: Accuracy percentages for intention-based ranking.

Figure 7 and Figure 8 show excerpt of response utterances in response class *reserve* and *other*, respectively. While response class *reserve* caters for utterances to confirm reservations, response class *other*, on the opposite, caters for utterances that do not contribute to the domain of conversation, including greeting and thanking.

[1] You have reserved “Dat heeft zo'n jongen toch niet nodig”, played by Herman Finkers on Sunday 28 May 1995. Commencement of show is 20:00. You are requested to collect these tickets minimum half an hour before commencement of the show.
[2] You have reserved “De Olifantsdracht”, played by Han Romer and Titus Tiel Groenestege on Sunday 22 January 1995. Commencement of show is 20:00. You are requested to collect these tickets minimum half an hour before commencement of the show.
[3] You have reserved 3 tickets for “Der zigeunerbaron”, played by Music Theater Prague on Saturday 11 March 1995. Commencement of show is 12.30. You are requested to collect these tickets minimum half an hour before commencement of the show.
[4] You have reserved 4 tickets for “Mevrouw Warrens Beroep”, played by The National Theater on Saturday 6 May 1995. Commencement of show is 20:00. You are requested to collect these tickets minimum half an hour before commencement of the show.

FIGURE 7: Response Utterances in Response Class *reserve*

[1] Do you want any further information?
[2] I have no information about this.
[3] Thanks for the effort and good-bye!
[4] With pleasure and Good Bye!

FIGURE 8: Response Utterances in Response Class *other*

Recall that intention-based ranking is performed based on the informativeness of the utterances; hence the interpretations of underlying semantics through domain attributes like *date*, *title*, and *other*. Albeit the variation of surface structures in both response utterances, observe that domain attributes almost all the time round down to the same attributes. For example in response class *reserve*, utterance [1] and [2] share the same set of domain attributes, which are *title*, *artist*, *date*, and *time*. Similarly, utterance [3] and [4] shares domain attributes of *ticket*, *title*, *artist*, *date*, and *time*. Assignment of domain attributes for response class *other* is no better because obviously the absence of semantics to the utterances forced the domain attribute *other*.

Since domain attributes are most likely the same in majority of the utterances, there are good chances that probabilities assigned by the ME model round down to the same figure. This leads to low accuracy results for ranking because our model is based on informativeness of utterances; hence the lack of it will average out the probability scores. At the end, one response utterances can hardly be weighed up or down from one another. Nonetheless, the impact of intention-based ranking through classification-and-ranking approach is proven to be superior to ranking based on surface features as presented by the techniques previously discussed. The ranking accuracies are consistent among the response classes, free from the influence of surface structure.

5. CONSLUSION & FUTURE WORK

Intention-based ranking differs from surface-based ranking in two major ways. Firstly, intention-based ranking apply a principled way to combine pragmatic interpretation of user utterance and the informativeness of the response utterance based on intentions, while surface-based ranking only attempts to find the best grammatical sequence of words that correspond to some meaning representations. Secondly, intention-based ranking is required to rank the utterances on the basis of relevance of a particular response utterance with regards to the input utterance. Surface-based ranking, on the other hand, is based on 'fluency' or 'completeness' of output sentences.

In the essence, as long as the technique relies on surface features [1, 5, 6], ranking accuracy is not uniform across the response classes, but rather dependent on surface representations of response utterances in individual response class. Due to this observation, in the future, we would like to investigate the performance of intention-based ranking model on other domain. Because our intention-based architecture assume the existence of dialogue act-annotated dialogue corpus based on DAMSL annotation scheme, we plan to use the MONROE corpus [15] that provides grounding and speech acts in order to run our comparative experiment.

6. REFERENCES

1. I. Langkilde and K. Knight. *"Generation that exploits corpus-based statistical knowledge"*. In Proceedings of the 36th Annual Meeting on Association for Computational Linguistics, Montreal, Quebec, Canada, 1998.
2. I. Langkilde. *"Forest-based statistical sentence generation"*. In Proceedings of the North American Meeting of the Association for Computational Lingustics, 2000.
3. I. Langkilde. *"HALOGEN: A Foundation for General-Purpose Natural Language Generation: Sentence Realization Using Probabilistic Models of Language"*. Ph.D. thesis, Information Science Institute, USC School of Engineering, 2002.
4. S. Bangalore and O. Rambow. *"Exploiting a probabilistic hierarchical model for generation"*. In Proceedings of the 18th International Conference on Computational Linguistics, Saarbrucken, Germany, 2000.
5. S. Varges. *"Instance-based Natural Language Generation"*. Ph.D. thesis, School of Informatics, University of Edinburgh, 2003.

6. A. Ratnaparkhi. *"Trainable approaches to surface natural language generation and their application to conversational dialogue systems"*. Computer, Speech & Language, 16(3):435–455, 2002.
7. A. Oh and A. Rudnicky. *"Stochastic natural language generation for spoken dialogue systems"*. Computer Speech & Language, pp. 387–407, 2002.
8. J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu and A. Stent. *"An architecture for a generic dialogue shell"*. Natural Language Engineering, 6(3), 2000.
9. A. Belz. *"Statistical generation: Three methods compared and evaluated"*. In Proceedings of the 10th European Workshop on Natural Language Generation, ENLG 05, 2005.
10. A. Mustapha, M. N. Sulaiman, R. Mahmod and M. H. Selamat. *"Classification-and-ranking architecture for response generation based on intentions".* International Journal of Computer Science and Network Security, 8(12):253–258, 2008.
11. M. White. *"Reining in CCG chart realization"*. In Proceedings of the 3rd International Natural Language Generation Conference, Hampshire, UK, 2004.
12. E. Velldal and S. Oepen. *"Maximum entropy model for realization ranking"*. In Proceedings of the 10th MT-Summit (X), Phuket, Thailand, 2005.
13. G.v.d. Hoeven, A. Andernach, S.v.d. Burgt, G-J. Kruijff, A. Nijholt, J. Schaake and F.d. Jong. *"SCHISMA: A natural language accessible theater information and booking system"*. In Proceedings of the 1st Internation Workshop on Applications of Natural Language to Data Bases, pp. 271-285, Versailles, France, 1995.
14. J. Hulstijn. *"Dialogue models for inquiry and transaction"*. Ph.D. thesis, University of Twente, Netherlands, 2000.
15. A. Stent. *"A conversation acts model for generating spoken dialogue contributions"*. Computer Speech and Language, 16(3):313–352, 2002.

Design of Cryptographically Strong Generator By Transforming Linearly Generated Sequences

Matthew N. Anyanwu manyanwu @memphis.edu
Department of Computer Science
The University of Memphis
Memphis, TN 38152, U.S.A.

Lih-Yuan Deng lihdeng @memphis.edu
Department of Computer Science
The University of Memphis
Memphis, TN 38152, U.S.A.

Dipankar Dasgupta ddasgupt @memphis.edu
Department of Computer Science
The University of Memphis
Memphis, TN 38152, U.S.A

Abstract

Random numbers have been used extensively in many simulation applications like Monte Carlo Integration or computer modeling. But recently security applications have increased the need for strong (secure) random number generation like automatic password generation, encryption algorithms, on-line gambling etc. Thus random number generation has become a challengingand an interesting task. Most classical random number generators, generate sequences that are either linear or predictable hence not suitable for cryptographic and security applications. Others generate sequences that even though they are secure they are not cryptographicallystrong and above all are slow in execution. Also recent advances in random number generation like the construction of Multiple Recursive Generator(MRG) with large orders, Fast Multiple Recursive Gener-ator (FMRG) and DX(system of multiple recursive generators proposed by Deng and Xu [2003]) generators does not generate a strong random number sequences. Though MRGs have extremely long period of length with good empirical performance, its recurrence equation can be solved given a small set of its generated sequence, this implies that MRGs and FMRGs are not strong cryptographic generators. We propose an algorithm that will transform linear sequences generated by both classical LCG, MRGs, FMRGs and DX generators and make them cryptographically strong generators by hiding the entire sequence generated by the generators, thus it will be difficult for cryptanalyst to predict or infer the generator sequence if even the partial sequence or the parameters or knowledge of the algorithm used in the transformation of the generators
are known.

Keywords: Linear Congruential Generator (LCG), Multiple Recursive Generator (MRG), Random Number generation, Strong (Secure) generators and classical generator

1. INTRODUCTION

The quality of any simulation application or study depends to a large extent the quality of random number generator. In the same manner the security of any cryptographic application that uses random numbers depends on the strong nature of the random number generators. Classical generators like LCGs generates random sequences that are linear, fast, simple and easy to implement, thus they have wide range of uses in simulation applications. But recently there have been great demand for random generators by security applications like automatic password generation, on-line gambling, digital signatures, encryption algorithms etc. A random generator is cryptographically strong if it is not possible to predict the value of the next random bit and also if it is generally incompressible (See, Bruce [1994]). The efficiency of a good random number generator for cryptographic purposes is determined by the number of mistakes in predicting the next random bit and also if the prediction can be computed in polynomial time (See, Hugo [1998]). Cryptographic generators are also expected to be fast in generating its output sequences, and the sequences it generates should be difficult to predict or infer. There have been recent advances in random number generation to generate sequences that are cryptographically strong but experimental results have shown that these generators are still insecure as their next bits can be predicated. Recent advances in random number generation include MRGs with large order (See, Deng [2005]), FMRGs (See, Deng and Lin [2000]) etc. In Section 2, we review some of the classical generators and state why they are not cryptographically strong generators by enumerating their shortcomings. In Section 3, we review the recent advances in random generation and attempt to make these generators strong cryptographic generators. We also stated how to transform these generators into strong generators. In Section 4, we proposed an algorithm that will transform the sequences generated by classical linear generators like LCGs and recent generators to produce a strong cryptographic generator that will be difficult for a cryptanalyst to predict or infer. Our algorithm transform the random bits generated by both linear classical generators and MRGs into a cryptographically strong bits by performing a transformation on the original generated bits and then perform a bitwise exclusive-or operation between the original bits and the transformed bits. The transformation of the bits is used in hiding the generated bit sequence from the attacker. The transformation also breaks the linear structure of the linear sequences making them difficult to predict. The final output are sequences that are difficult to predict and strong for cryptographic purpose. In Section 5, we did a comparison between sequences generated by classical generators and our transformed sequences.

2. Classical Random Number Generators

Pseudo-random number generators (PRNG) are random number generators that produce sequence of values based on an initial seed and current state. They are called deterministic generators in that given the same initial seed they will output the same sequence values. Random numbers have been applied in simulation experiments, randomized algorithms, and Monte Carlos methods in numerical analysis and also in cryptography (See, Neal [1994]). PRNGs used for cryptographic purposes are expected to be fast (computable in polynomial time) and secure (See, Stinson [2006]). But classical generators do not meet these two objectives simultaneously. Most classical generators like RSA and Blum-Blum-Shub (BBS) which are called power generators are cryptographically secure but too slow and cumbersome in performance for cryptographic applications. Other classical generators based on linear congruencies like Linear Congruential Generator (LCG) or linear feedback shift registers (LFSR) are very fast and easy to implement. But these linear generators are not cryptographically secure (the next bit stream value generated can be predicted), thus they are not good for cryptographic applications A PRNG is cryptographically secure if it is computationally infeasible to predict the value of the next random bit, even with knowledge of the algorithm, the hardware that generates the sequence and also the previous bit
streams (See, Bruce [1994]). Also secure PRNG are generally incompressible (See, Bruce [1994]), except when given the key (algorithm). In this section we will review some early classical

generators like RSA, BBS, BM (Blum Micali), LCG, LFSR and recent advances in pseudo-random number generators (Multiple Recursive Generators (MRG)

2.1 Linear Congruential Generator:

Linear Congruential Generator (LCG) was proposed by Lehmer [1951]. It uses a linear function over modulus arithmetic field. LCG is one of the most widely used PRNG, especially in simulation applications. It is produced iteratively as shown in the equation below;

$$X_i = (BX_{i-1} + c) \bmod m, \quad i \geq 1, \quad (1)$$

where Xi, B, c, and m are positive integers and are called the parameters of LCG (Xi are all integers between 0 and m-1). The quality of the generator depends on the selection of the increment c, multiplier B, initial seed X0, and modulus m. If c =0, the LCG is called mulplicative linear congruential generator (MLCG) as shown in the equation below;

$$X_i = (BX_{i-1}) \bmod m, \quad i \geq 1, \qquad (2)$$

The number sequence generated by the LCG is of the form X0, X1, X2... , where X0 is the initial seed. A series of researches and experiments have shown that LCG sequences can be easily predicted and this raises some doubt about the security of LCG when it is used in any cryptosystem, thus it is not suitable for cryptographic applications.Boyar [1982] and Bruce [1994] showed that if the whole sequence of LGC is published, the sequence can be predicted if the initial seed is given, even though other parameters (B, c or m) of LCG may be unknown. LCGs have been broken as specified in articles (See, Boyar [1989] and Reeds [1977]). Thus LCGs are not good cryptographic generator but very useful in non-cryptographic applications like simulation. An on-line poker game that uses random numbers generated by LCG to shuffle deck of cards will be flawed because the LCG sequences can be easily be predicted and this will lead to cheating in the game if the sequences generated by LCG are not made secured.

2.1.1 Predicting LCG Generator

Boyar [1982] used the Plumstead's algorithm to predict the LCG sequence using partial consecutive sequence of the generator, the number of the consecutive sequence used depends of the size of the modulus. In simulation of the Plumstead's algorithm by Pommerening (See, Pommerening [2003]), when the size of the modulus is of the order of (231 − 1), ten consecutive sequence of the generator are needed for the prediction of LCG sequence. There are also many other methods of predicting the sequence of LCG which depends on LCG parameters given. If the multiplier (B) is unknown but the partial sequence and the modulus are known then the LCG (2) can be predicted by calculating the modulus inverse as stated below; B=X1*X0-1 mod m,Where X1, X0 are the consecutive variates of the LCG. The value of the multiplier B will then be substituted to predict the LCG sequence (2). If only the partial sequence of LCG generator is given, the modulus, multiplier and the seed of the LCG can be determined using Haldir method of cracking LCG generator (See, Haldir [2004]). Haldir cracking of LCG generator is based on George Marsaglia analysis of pseudorandom random number generators. Marsaglia pointed out that there is a flaw in pseudorandom generators as their uniform variants when viewed as points in a cube fall into hyper-plane, which contains points that are well distributed in a unit cube. The LCG sequence is used in forming a matrix, the determinant of the matrix will give an integer multiple of the modulus m. Then the gcd of a number of matrices gives the actual value of the matrix. Given the LCG as in (1) where c =0, if X0, X1, X2... are observed from the sequence of LCG, then they can be used to
form a matrix thus;.

$$\begin{matrix} X1 & X2 \\ X2 & X3 \end{matrix}$$

The determinant $D = X1 \times X3 - X2 \times X2$. Note that

D mod m = 0(since D is an integer multiple of modulus m. Thus modulus m = GCD (D, for some number of matrices (about 4 or 5).
If the LCG is as in (1) where there is a constant, the matrix formed is as given below using Haldir cracking of LCG generator method as shown below;

$$\begin{array}{ccc} X1 & X2 & 1 \\ X2 & X3 & 1 \\ X3 & X4 & 1 \end{array}$$

Given the LCG sequence of consecutive numbers; 207482415, 1790989824, 2035175616, 77048696, 24794531,

109854999, 1644515420, 1256127050. Using Haldir method, the LCG sequence is used to form a 3 × 3 matrix of

the form;

$$\begin{array}{cc} 207482415 & 1790989824 \\ 1790989824 & 2035175616 \\ 2035175616 & 77048696 \end{array}$$

$$\begin{array}{c} 1 \\ 1 \\ 1 \end{array}$$

The implementation of Haldir's method using the LCG sequence predicts the LCG sequence thus; m =2147483647, k= 16807 and seed =12345.
When we applied Haldir's method to our proposed algorithm(as in section 4), the transformed LCG sequence is not predicted, because the transformed LCG sequence are hidden by applying our proposed algorithm in section 4, making it computationally difficult for an attacker to infer the sequence of an transformed LCG sequence. Thus knowing the partial sequence or some of the parameters of the transformed LCG generator does not pose a threat any longer as our proposed algorithm is applied to the transformed LCG sequence.

2.2 Linear Feedback Shift Register (LFSR) Generator:

LFSR is used in generating sequence of binary bits. It is a PRNG whose input bit is a linear function of two or more bits. It uses a short seed to produce sequence of bits which has a very long cycle. LFSR can be said to be a special type of Multiple Recursive Generator (MRG) but it differs from MRG in that it generates binary sequence of bits while MRG generates sequence of numbers. It is based on the linear recurrence which is of the form stated below (See, Tausworthe [1965]);

$$X_i = (\alpha_1 X_{i-1} + \cdots + \alpha_k X_{i-k}) \bmod 2, \qquad (3)$$

The linear recurrence of LFSR is based on Z2 recurrence. The sequence of values produced by LFSR is determined by its current or previous state. The parameters of LFSR are k > 1 (the order of the recurrence), Xi (bit generated), α (the multiplier) and X0 the initial seed. The maximum period of the recurrence is 2k – 1, if and only if the characteristics polynomial is as given below;

$$F(x) = 1 + \alpha 1x + \alpha 2x2 + \cdots + xk \quad (4)$$

The characteristic polynomial (4) is a primitive polynomial in Z2, which is a finite field with 2 elements (See, Lidl [1994]). The recurrence of (3) shows how each bit of the state evolves and also that each bit of the output is a linear combination in Z2 of the given state (See, Tausworthe [1965]). LFSR like LCG is a fast generator; it produces linear sequence of bits. LFSR sequence like that of MRG can be predicted as stated in section 3.1.1. Also LFSR bit-string sequence can be predicted using Berlekamp-Massey algorithm (See, Massey [1969] and Berlekamp [1968]).

2.3 Blum-Micali (BM) Generator

BM is a cryptographically secure generator presented by Blum and Micali [1982]. It is based on the difficulty in computing discrete logarithms [See, Stinson 2006], which is also based on the believe that the modular exponentiation modulo is prime and a one-way function. Thus it is based on the underlying discrete logarithm over a finite field and also on the assumption that solving discrete logarithm problems is a hard problem even when
the exponent is small.
The general form of BM is as given below;

$$Xi+1 = aXi \bmod m, \quad i \geq 0 \qquad (5)$$

The output of the generator is 1 if Xi < m/2, otherwise 0. The constant a is a primitive root in the finite field.
This generator is secure if the modulus m is very large (1024-bit modulus) thus it is computationally difficult to compute the discrete logarithms mod m. BM is not suitable for cryptographic applications even though it is secure, because the generation of the output sequence is very slow.

2.4 Blum Blum Shub(BBS) Generator:

Blum, Blum and Shub [1986] invented the BBS pseudo-random number generator. BBS is based on the hardnessof quadratic residues and inter factorization over modulus field (See, Stinson [2006], Raja and Hole [2007]). Itis a simple and secure cryptographic generator. BBS is generally of the form;

$$Xi+1 = (Xi)2 \bmod m, \quad i \geq 1 \qquad (6)$$

$$Zi = Xi \bmod 2 \quad (7)$$

Zi is the output of the generator.

The modulus m of BBS is the blum integer. The relationship between the modulus m and the two large prime integers p and q is given below.
$m=p\times q$ and $p\equiv q\equiv 3 \bmod m$.

The security of this generator rests on the ability of the cryptanalyst to factor m. Also only one bit of the sequence is generated at a time instead of the whole number (See, Boyar [1989]), this makes the generator very slow in generating its sequences. Also it cannot be used as a strong cryptographic generator because it has been shown that in the Blum protocol of BBS, the modulo m can be factored during key exchanges (See, Hastad and
Shamir [1985]).

2.5 RSA Generator

RSA generator is based on the assumed security of RSA cryptosystem proposed by Rivest, Shamir, and Adleman[1978]. It is based on the hardness of integer factorization of the modulus number (See, Stinson [2006]). RSA forms a sequence of numbers in which each element in the sequence is an RSA encryption of the previous element. The least bit of the element forms the bit-string. The general form of RSA is as stated below;

$$Xi+1 = Xie \bmod m, \quad i \geq 1 \qquad (8)$$

$$Zi = Xi \bmod 2 \qquad (9)$$

Zi is the output of the generator.
The modulus m is a product of two large prime numbers p and q (512-bit primes), e is chosen so that

$$\gcd(e,\varphi(m)) = 1 \ (10)$$

where m and e are public key while p and q are the secret keys. The security of RSA is based on finding the factors of m, If m is large enough(1024-bit modulus) it will be difficult to factor it, RSA is a very slow generator, as only one bit of the sequence is generated at a time instead of the whole number.

3. Recent Advances in Random Number Generation

The classical generators like BBS, BM, RSA, LCG and LFSR described above are not good candidates for security applications. While BBS, BM and RSA are secure, they are slow in generating their bit sequences yet they are being used in some security applications as they are readily available. LCG and LFSR are not secure but they are fast in generating their sequences and also easy to implement. Cryptographically secure generators are supposed to be fast in generating its sequences and also its bit sequence is supposed to be difficult to predict

3.1 Multiple Recursive Generators (MRGs)

MRG is one the most extensively studied and most widely used PRNG which is based on the k-th order linear recurrence. The next value of MRG is computed recursively based on its previous values (See, Lehmer [1951]). The basic equation of MRG is thus stated below;

$Xi = (\alpha 1Xi-1 + \cdots + \alpha kXi-k) \bmod m, i \geq k(11)$
The starting values are (X0, X1, X2... Xk−1), which are not all zero; m is the modulus which is a large prime number and Xi can be transformed using Ui = Xi/m. In order not to obtain 0 or 1, (See, Deng and Xu [2003]) transformed
Ui = (Xi +0.5)/m. The parameters of MRG are Xi (generated sequence), αk(the multiplier), m(the modulus)and the order k. The maximum period of an MRG of an order k is mk − 1, which is obtained if the characteristic polynomial is as given below;

$f(x) = xk - \alpha 1xk-1 - \cdots - \alpha k, \qquad (12)$

is a primitive polynomial and the modulus is prime. MRG is an improvement of LCG as it has good empirical performance, long periods and higher dimensional uniformity when compared to LCG. MRG reduces to LCG when its order k=1. When the order k of MRG is large, it becomes less efficient as it needs several multiplications compared to LCG which needs only one multiplication. To improve the efficiency of MRG many authors and researchers proposed improvements to it. Deng and Lin [2000] proposed FMRG which needs only one multiplication like LCG, thus in addition to features inherent in MRGs, FMRG is very fast, making it as efficient as LCG. Deng and Xu [2003] and Deng [2005] improved the performance of FMRG by proposing a system of DX generators which are fast, portable and efficient with maximal period compared with FMRG. They fixed the coefficient of the non-zero multipliers to be equal. The DX (DX-k-s(s =1 t4)) generators by
Deng and Xu [2003] are as stated below:

1. DX-k-1 [FMRG] ($\alpha 1 = 1, \alpha k = B$).

$$Xi = Xi-1 + BXi-k \bmod m, \quad i \geq k.(13)$$

2. DX-k-2 ($\alpha 1 = \alpha k = B$).

$$Xi = B (Xi-1 + Xi-k) \bmod m, \quad i \geq k. \qquad (14)$$

3. DX-k-3 ($\alpha 1 = \alpha\lceil k/2\rceil = \alpha k = B$).

$$Xi = B (Xi-1 + Xi-\lceil k/2\rceil + Xi-k) \bmod m, \quad i \geq k. \quad (15)$$

4. DX-k-4 ($\alpha 1 = \alpha k/3 = \alpha\lceil 2k/3\rceil = \alpha k = B$).

$$Xi = B (Xi-1 + Xi-\lceil k/3 + Xi-\lceil 2k/3\rceil + Xi-k) \bmod m, \quad i \geq k. \qquad (16)$$

3.1.1 Predicting MRG Generator

The characteristics equation of MRG as defined in (12) has been determined by the first 2k terms of the sequence using a system of k equations (See, Lidl and Niederreiter [1994]). MRG sequence with the modulus and the partial sequence known can also be predicted, when the system of linear equations which can be formed using the recurrence of (11) is solved (See, Stinson [2006]) as shown below;

- $\alpha 1X1 + \alpha 2X2 + \cdots + \alpha kXk = Xk+1 \bmod m$
- $\alpha 1X2 + \alpha 2X3 + \cdots + \alpha kXk+1 = Xk+2 \bmod m$
- $\vdots$
- $\alpha 1Xk + \alpha 2Xk+1 + \cdots + \alpha kX2k-1 = X2k \bmod m$

These systems of linear equations can be re-written in matrix format as

Xk+1

Xk+2 =

X2k

X1 X2 ••• Xk α1

α2

X2 X3 ••• Xk+1

.

.

.

Xk Xk+1 ••• X2k−1 αk

The above equation can also be re-written as X (column), α (column) and M (matrix) as shown below,X = M* α (mod m). (17)This gives rise to k equations in k unknowns. If the matrix M (k × k) in (17) has an inverse (mod m), then the solution to (17) can be obtained thus; α= M−1* X (mod m). (18)The values of the multipliers (α1, α2, •••, αk) can then be obtained by solving (18). Hence this will lead to the prediction of MRG sequence of (11), given the modulus m. If the modulus of MRG is not known but the partial sequence (Xk+1,Xk+2,Xk+3,...,X2k) (at least 2k terms) for k-th order MRG is given, the sequence of the MRG can be predicted by forming a matrix of (k + 1) rows using consecutive terms of the sequence as done in the case of LCG. The determinant D of the matrix can then be calculatedcalculated is an integer multiple of the modulus m, thus D (mod m) =0. The actual modulus is calculated thus m = gcd (D, for a number of matrices (about 4 -5)). The sequence of the MRG is then used to form a matrix equation; the modulus is substituted and used in solving the equation of (18) to calculate other parameters of the MRG. Given the following MRG sequences; 1870519315, 618460120, 287572083, 2134648799, 83050304, 1713934911, 1872854851 with initial seeds of X0 = 12345678, X1 = 207482415, X2 = 79098924 and modulus m = 2147483647, of k-th order =3, we used (17) and (18) to obtain the parameters of the MRG and hence predict its sequence as
shown below;

2134648799

83050304
1713934911

1870519315

=

618460120

287572083

618460120	287572083	α1
287572083	2134648799	α2
2134648799	83050304	α3

The solution to the above matrix equation in modulus m (we developed a c-code using NTL (Number Theory Library) packages to solve the matrix equation) gives the parameters (multipliers) of the MRG, the solution is as
stated below;
(α1, α2, α3) = (16807, 1036675, 1047849).
The MRG sequence can be predicted by substituting the parameters obtained in (11), which will give the matrix equation as

Xi = (16807Xi−1 + 103667Xi−2 + 1047849Xi−3) mod 2147483647, i ≥ 3 (19)
where X0 = 12345678, X1 = 207482415, X2 = 79098924. Using (19), we predicted the sequence of the MRG.
When this method of obtaining the parameters of MRG is appliedto our proposed algorithm (section 4) for the MRG, the parameters of the MRG was not obtained because our proposed algorithm hides the MRG sequence making it difficult for an attacker to infer or predict the sequence. Thus knowing the partial sequence or any of the parameters of the MRG does not pose a threat any longer when our proposed algorithm is applied.

4. Algorithm for secure and efficient pseudo-random number generator

In order to make LCGs and MRGs and other linear PRNG strong cryptographic generators. We propose an algorithm that will transform the linear sequence generated by the PRNGs. The transformation (20) hides the generated bits sequence, breaks the linear structure of the sequences and then perform a bitwise exclusive OR between the original bits and the transformed bits

The output of our algorithm gives a bit sequence which will be practically impossible for an attacker to infer in computational time. Our algorithm is stated below;

Yi = Xi T (Xi+1). (20)
In transforming the generated bits, we first truncate the bit performing a binary shift operation as shown
below;
STEP 1: A transformation is performed on the sequence (Xi)

Xi = (b1, b2... bn) mod 2 (21)
The transformation produces T (Xi+1) given below;
T (Xi+1) = (bn, bn−1... b2, b1) mod 2. (22)
STEP 2: A bitwise exclusive-or operation is then performed between the transformed and the original generated sequences as shown below;

Yi = Xi T (Xi+1), (23)

where denotes exclusive OR operation between the transformed sequence and the original generated sequence. The exclusive OR operation is used in producing a secure cryptosystem as explained in the next subsection. The final output of our algorithm (23) gives a cryptographically strong pseudorandom sequence which is computationally difficult to infer or predict. The transformed sequence hides the bits for each of the generated output sequence and breaks its linear structure making it impossible for the cryptanalyst to infer or predict the linear equence or solve the characteristics equation of the MRGs by a system of k equations Our algorithm not only hides the originally generated bits but it performs exclusive OR operation between the transformed bits and originalbits. The exclusive OR operation is necessary to ensure that the final output sequence is computationally
difficult to infer.

4.1 Simulation of the Proposed Algorithm

We applied our algorithm as stated in (23) to equations (2), (13), (14), (15) and (16). We implemented our algorithm using c-code supplements, various parameters of both LCG and MRGs were used. The algorithm is based on a 32-bit binary sequence. Our algorithm produces a secure generator that is very fast and computationally difficult to infer or predict, this is evidence by the following features;

• Our algorithm generates its bits sequence in polynomial computable time; the generation of the bits is also very fast.
• We used linear sequences (LCGs, MRGs) by Deng [2005, 2008] that passed standard statistical tests, thus both the input and output bit-strings of the algorithm are well distributed.
• The algorithm uses initial seed of short sequences and generates output long sequences bit-strings that are well distributed.
• Our algorithm hides the entire bits of the bit-string generated by linear generators, making it computationally difficult for an attacker to infer or predict the bit string of the generator.

Our Algorithm hides the bits generated by the linear generators using the following techniques;
• Our algorithm transforms the bit-string sequence by reversing the order of the bits of the bit-string after
truncating its lower bits before performing a Bitwise Exclusive-OR operation on the bits.
• The reversal of the bit-string sequence by our algorithm breaks the linear structure of the linear sequences
producing a non-linear sequence. Also non-linear sequence is more secure when compared with linear
structure as evidenced by BBS, BM and RSA generators
• The Bitwise Exclusive-OR between the transformed bit-string and the original bit-string by our algorithm
is used to toggle the positions of bits in the original bit-string with the transformed bit-string making it
computationally difficult for the attacker to predict correctly the position of the bits in the original bit-string.
• The Bitwise Exclusive-OR between the transformed bit-string and original bit-string is used to produce a
cipher bit-sting. Thus when an attacker attempts to predict the bit-string he will only succeed in getting
the cipher bit and not the original bit. This makes the bit-string produced by our algorithm to be computationally difficult to predict
• The Bitwise Exclusive-OR operation introduces a piling-up lemma (See, Lemma 3.1 of Stinson [2006, page

81]), which ensures that the binary variables are independent, that is the state of one has no effect on the
state of any of the others which also ensures equi-distribution among the binary bits. Thus it is difficult for
an attacker to predict the transformed even if some part of the sequence is revealed.
Thus our algorithm produces a strong generator by transforming the linear sequences produced by LCG and MRGs into unpredictable sequence of bits.

4.2 Security of the Proposed Algorithm

The figures below shows that the sequence generated by linear generators is linear in structure (figure 1) while our transformed sequence is not linear (figure 2). We used LCG (B, m) to denote LCG (2) with multiplier B and prime modulus m. Thus we used LCG (B=3, m=127) to show that LCG sequence is linear (figure1), while the transformed LCG (B=3, m=127) is not linear. Figure 2 shows that our algorithm breaks the linear structure of the linear generators sequences, thus making them more difficult to infer or predict. The linear structure of the linear sequences is broken when our algorithm is applied by reversing the order of the bits, shifting the bits by one position to the right, and performing exclusive-OR operation between the original bit sequence and the transformed bit sequence. Also the same result in figure 1 and figure 2 will be obtained if MRGs and MRG ransformed by our algorithm is used. The security property of our proposed algorithm is based on the reversal of the bits sequence and the bitwise exclusive-OR operation between the original bit sequenceand the reversed bit sequence. Our proposed algorithm has transformed the linear sequence of LCG and MRG into non-linear sequence. Thus a hacker can obtain information about the parameters of LCG and MRG generators as stated above(for LCG generator) and given in the Table 1(for MRG generator)and attempt to used it to break our proposed algorithm but our algorithm hides the entire sequence generated by the linear generators making it difficult for him to crack the algorithm, hence knowing some of the partial sequence or the parameters of the linear generators LCG(B, c and m) and MRG(K, αk, and m) will no longer pose a security threat to the generators, as they cannot be used to solve for or predict the sequence of the generators. This makes our algorithm difficult to infer by attackers. Also if the hacker has information about the algorithm we used in transforming the generator sequences, it will be difficult for him to reverse (invert) the sequences back to linear sequence. Thus the transformed sequence now behaves like a non-linear sequence, which is difficult to infer or break.

4.3 Recommended Parameters for Our Proposed Generator

The modulus of the generators (LCG and MRG) transformed by our proposed algorithm and the one we have been using for our experiments is rather too small to be used for practical purposes. Park and Miller [1988] published minimal standard for LCG parameters. The minimal standards are large prime numbers of modulus m =231 − 1, B=16807 and c (constant term) =0 for 32-bit CPUs. We recommend this minimal standard for LCG parameters when our proposed algorithm is applied. Table1 below shows some of the parameters of DX generators by Deng [2008] determined by efficient search algorithm. We recommend the parameters in table 1 for our proposed algorithm used in transforming the sequence of MRG (DX) generators. The parameters in table 1 are used in producing efficient and portable multiple recursive generators of large order with good empirical performance and a period of approximately 1093384 (See, Deng [2008]).

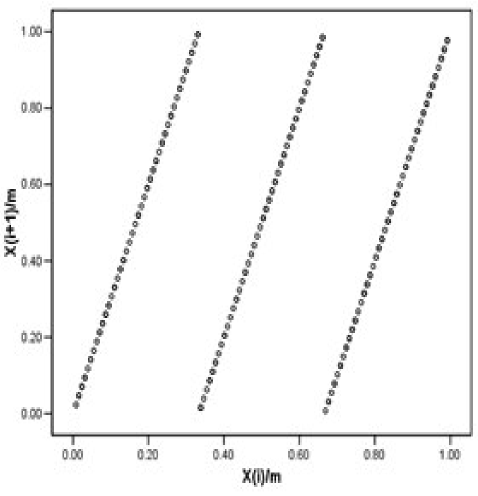


Fig.1: X(i+1)/m vs X(i)/m for LCG(B=3,m=127)

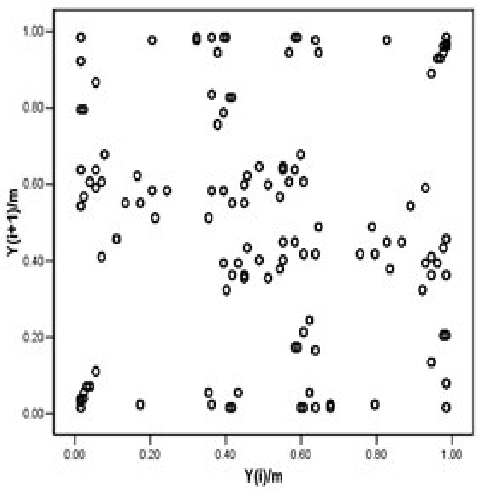


Fig.2: Y(i+1)/m vs Y(i)/m for the trasnformed LCG(B=3,m=127)

Table 1: List of k, m and $\alpha_{<}2^{30}$ for DX-k-s generators

k	m	s=1	s=2	s=3	s=4
5003	2146224359	1073727083	1073741516	1073730698	1073740466
6007	2137498943	1073738651	1073715261	1073729141	1073738504
7001	2146873559	1073709808	1073728419	1073738188	1073735327
8009	2142326903	1073717208	1073726014	1073733016	1073719175
9001	2140247399	1073737583	1073717540	1073733156	1073732451
10007	2147051903	1073726195	1073702542	1073723329	1073730725

5. Comparison between Classical Generators and Transformed Generators

As stated in proceeding sections classical generators, that are fast in generating their output sequences like LCGs are not cryptographically strong since their sequences can be predicted. Also classical generators that are secure are not suitable for cryptographic applications as they are very slow in generating their output sequences since the generators produces one bit at a time rather than the entire sequence at each step of the generation (See, Boyar [1989]) but they are still been used in security applications they are readily available. Classical generators like LCG can be easily predicted if the previous elements of the sequence and the initial value are given, without knowing the modulus and the multiplier (See, Hugo [1999]). LCG can also be predicted if the modulus and the previous sequence is given even though the multiplier and the initial seed are not known by using the inverse of the modulus (See, Frieze and Langarias [1984]). MRGs, FMRGs including DX generators can be predicted if the characteristics polynomial equation can be solved using a system of k equations (See, Lidl and Niederreiter [1994]). The BBS, BM and RSA generators are secure but they are not very fast in generating their bits sequence, the output sequence is generated one bit at a time . Most classical generators are linear generators in that the structure of the generated sequences is too regular and have a coarse lattice structure (See, Takashi et al [1996]). But the transformed sequence by our algorithm is fast, efficient, non-linear and strong, also difficult to predict or infer.

6. Conclusion

Experimental results and tests have shown that classical generators like LCGs that generate pseudorandom linear sequences are not suitable for cryptographic purposes, even though it is simple, efficient and easy to generate. Other classical generators like BBS, RSA, and BM etc that are thought to be secure are equally not good enough for cryptographic purposes as they are slow in generating the next random bit sequence. Also the recent advances in random number generation (MRGs and FMRGs) are fast and efficient in generating linear sequences with long periods and good empirical performance, but still they are not cryptographically strong as the linear system can be predicated using a system of unique k equations. Our proposed algorithm produces a strong pseudorandom sequence that is suitable for cryptographic purposes and difficult to predict/infer by transforming the linear sequences and breaking its linear structure. The transformation hides the linear bits of the generated linear sequence preventing the attacker from accessing the generated output sequence, even with the knowledge of the partial sequence, parameters of the generators and the algorithm used in transforming the generator sequence. Thus knowing the parameters and partial sequence of the generators does not pose any threat any longer as the prediction of the generator sequence will no longer be an easy one. Also the knowledge of the transformation algorithm will not be of much use to the attacker as he will find it difficult to reverse (invert) the transformed sequence back to linear sequence. The bits sequence generated by our algorithm has features described in section 4.2, which makes the output sequence to be fast, strong and also difficult to predict or infer.

References:

1. Alexi, W., Chor, B. Z.,Goldreich, O. and Schnorr, C.P. (1988). RSA and Rabin Functions: Certain Parts Are as Hard as the Whole Proceedings of the 25th IEEE Symposium on the Foundations of Computer
Science 449-457.

2.Berlekamp, E.R. 1968. Algebraic coding theory McGraw-Hill, New York, (1968).

3.Blum, M., and Micali, S. (1982). How to generate cryptographically strong sequences of pseudo-randombits.In Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science. IEEE, New York.
pp. 112- l 17

4.Blum, L., Blum, M., and Shub, M. (1983). A simple secure pseudo-random number generator. InAdvances in Cryptography: Proceedings of CRYPTO 82. Plenum Press, New York, 61-78.

5.Blum, L., Blum, M., and Shub, M. (1986). A simple unpredictable pseudo-random number generator.
SIAM Journal on Computing, Volume 15, Issue 2, pp. 364-383 . ISSN:0097-5397.

6.Boyar, J. (1982). Inferring sequences generated by a linear congruence. Proceedings of the 23rd AnnualIEEE Symposium on the Foundations of Computer Science pages 153-159.

7.Boyar, J. (1989). Inferring sequences produced by pseudo-random number generators. Journal of theAssociation for Computing Machinery 36,129-141.

8.Bruce, S. (1994). Applied Cryptography: Protocols, Algorithms and Source Code in C John-Willey andSons, New York, 1994. ISBN: 0-471-5975602

9.Deng, L. Y., and Lin, D. K. J. (2000). Random number generation for the new century. AmericanStatistician 54, 145-150.

10. Deng, L. Y. and Xu, H. Q. (2003). A System of High-dimensional, Efficient, Long-cycle and Portable Uniform Random Number Generators, ACM Transactions on Modeling and Computer Simulation, Vol. 13,
No. 4, pages 299-309.
11. Deng, L. Y. (2004). Generalized Mersenne prime number and its application to random number generation, in Monte Carlo and Quasi-Monte Carlo Methods 2002 (H. Niederreiter, ed.), Springer-Verlag, 167-180.
12. Deng, L. Y. (2005). Efficient and Portable Multiple Recursive Generators of Large Order, ACM Transac-
tions on Modeling and Computer Simulation(TOMACS), Volume 15, Issue 1 PP.1-13. ISSN: 1049-3301
13. Deng, L. Y. (2008). Issues on Computer Search for a Large Order Multiple Recursive Generators, in Monte Carlo and Quasi-Monte Carlo and Quasi-Monte Cralo Methods 2006(S. Heinrich, A. Keller and H. Niederreiter, ed.) Springer-Verlag, 251-261.

14. Frieze, A.M., Kannan, R.and Lagarias, J.C. (1984). Linear Congruential Generators do not Produce Random Sequences. 25th Annual Symposium on Foundations of Computer Science, pp. 480-484.
15. Hastad, J. and Shamir, A. (1985). The cryptographic security of truncated linearly related variables. In Proceedings of the 17th ACM Symposium on Theory of Computing (Providence, R.I., May 6-8). ACM, New York. 356- 362.

16. Haldir, R. (2004). How to crack a Linear Congruential Generator. http://www.reteam.org/papers/e59.pdf.
17. Hugo, K. (1999). How to Predict Congruential Generators. In Proceedings on Advances in cryptology(Santa Barbara California USA). 138-153. ISBN: 0-387-97317-6.
18. Lehmer, D. H. (1951). Mathematical methods in large-scale computing units. Proceedings of the Second Symposium on Large Scale Digital Computing Machinery, Harvard University Press, Cambridge, MA, 141-146.

19. L′ Ecuyer, P. (1997). Uniform Random Number Generation: A Review. Proceedings of the 29th conference on Winter Simulation conference, Atlanta Gergia USA 127-134.
20. L′ Ecuyer, P., Blouin, F., and Couture R. (1993). A search for good multiple recursive linear random number generators. ACM Transactions on Modeling and Computer Simulation, 3, 87-98.

21.Lidl, R., and Niederreiter, H. (1994). Introduction to Finite Fields and Their Applications. RevisedEdition. Cambridge University Press, Cambridge, MA.

22.Marsaglia, G. (1968). Random Numbers fall mainly in the planes. Proceedings of the National Academy of sciencesVol. IT-15, January 1969

23.Massey, J.L. (1969). Shift-register synthesis and BCH decoding. IEEE Trans. on Inform. Theory, 61,25- 28.

24.Neal R. W. (1994). The Laws of Cryptography, online book http://www.cs.utsa.edu/wagner/lawsbookcolor/laws.pdf

25.Park, K. and Miller, K. 1988. Random Numbers Generators: Good Ones are Hard to Find. Communication of the ACMvol. 31.No.10, pp 1192-1201

26.Pommerening, K. (2003). Prediction of linear congruential generators http: //www.staff.uni–mainz.de/pommeren/

27.Raja,G and Hole, P.H. (2007). Use of the Shrinking Generator in Lightweight Cryptography for RFIDhttp://www.autoidlabs.org/uploads/media/AUTOIDLABS-WP-SWNET-024.pdf

28.Rivest, R., Shamir, A., and Adleman, L .(1978). A method for Obtaining Digital Signatures andPublic-Key Cryptosystems. Communications of the ACM, 21, 120-126.

29. Reeds, J.A (1977). Cracking Random Number Generator. Cryptologia, v.1, n. 1, 20-26
30.Shamir, A. (1983). On the Generation of Crytographically Strong. ACM Transaction on Computer Sys-tems (TOCS), pp. 38-44, Volume1 Issue1. ISSN: 0734-2071
31. Stinson, D. (2006). Cryptography: Theory and Pratice 3rd edition CRC Press, Boca Raton, Florida, ISBN:
1-58488-508-4.

32. Takashi, K., Li-Ming, W. and Niro, Y. (1996). On a nonlinear congruential pseudorandom number generator Mathematics of Computation, Volume 65, Issue 213, 227 - 233, ISSN: 0025-5718 .

33. Tausworthe, R.C. (1965). Random numbers generated by linear recurrence modulo two Mathematics ofComputation, 19: 201-209.

Managing Software Change Request Process: Temporal Data Approach

A. R. M. Nordin mohdnabd@udm.edu.my
Faculty of Informatics
Universiti Darul Iman Malaysia, KUSZA Campus
21300 K Terengganu, Malaysia

S. Suhailan mohdnabd@udm.edu.my
Faculty of Informatics
Universiti Darul Iman Malaysia, KUSZA Campus
21300 K Terengganu, Malaysia

Abstract

Change request management is the process that approves and schedules the change to ensure the correct level of notification and minimal user impact. Complexity level of this process would become complicated should software was distributed in many places. There are several techniques and tools have been produced to perform change request management. But most of these techniques do not give enough attention towards dynamic aspect of information management such as temporal base elements for the associated data. Therefore, this paper presents a new dimension to support software change request management. Temporal elements such as valid time and transaction time are the main attributes considered, to be inserted into the software change request management system database. By inserting these elements it would help the people involved in change request management to organize data and perform activity monitoring with more efficient.

Keywords: change request, temporal database, valid time, transaction time.

1. INTRODUCTION

Temporal based data management has been a hot topic in the database research community since the last couple of decades. Due to this effort, a large infrastructure such as data models, query languages and index structures has been developed for the management of data involving time. Presently, many information systems have adopted the concepts of temporal database management such as geographic information systems and artificial intelligence systems [1][3][6][7]. Temporal management aspects of any objects transaction data could include:

- The capability to detect change such as the amount of change in a specific project or object over a certain period of time.
- The use of data to conduct analysis of past events e.g., the change of valid time for the project or version due to any event.
- To keep track of all the transactions status on the project or object life cycle.

Change request is a formally submits artefact that is used to track all stakeholder requests including new features, enhancement requests, defects and changes in requirement along with related status information throughout the project lifecycle. All change history will be maintained with the change request, including all state changes along with dates and reasons for the change. This information will be available for any repeat reviews and for final closing.

Meanwhile, software change request management is one of the main requirements to achieve quality in software maintenance process. Change request managers must devise a systematic procedure to ensure that all the change requests has been performed by the maintainer in required time span and fulfil with the user requirements. To achieve this goal the managers need to introduce mechanism to handle requests evaluation, and authorization of changes. In addition, the manager also collects statistics about components in the software system, such as information which determining problematic components in the system.

In this paper, we will introduce the technique to improve software change request management and monitoring model using temporal elements such as valid time, transaction time and temporal operators. For this purpose, these temporal elements will be stamped into each activity involve in a change request management. The model develops a database that maintains the changes occurred to the software change request information. Therefore, it provides a historical data retrieval platform for an efficient software maintenance project. In Section 2, the concept of temporal data management is reviewed. Section 3 briefly discusses the current approach of software change request process. The proposed model of temporal based software change request process is discussed in Section 4. Conclusion will be discussed in Section 5.

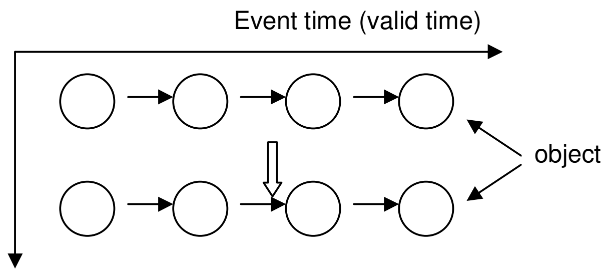
2. TEMPORAL DATABASE MANAGEMENT

Most of the data models maintain only concept of current view data which an existing data value is overwritten by a new incoming value during the updating process [12]. To overcome this problem, temporal databases capture the evolution of the modelled reality over time by maintaining many data states e.g., past, current and future. To support this, the use of time stamping is proposed [7]. To date, two well-known time elements which are usually considered in the literature of temporal database are transaction time and valid time.

The valid time of a database fact is the time when the fact is true in the *miniworld.* In other words, valid time concerns with the evaluation of data in respect to the application reality that data describe. Valid time can be represented with single chronon identifiers (e.g., event time-stamps), with intervals (e.g., as interval time-stamps), or as valid time elements, which are finite sets of intervals [6][7]. Meanwhile, the transaction time of a database fact is the time when the fact is current in the database and may be retrieved. It means, the transaction time is the evaluation time of data with respect to the system where data are stored. Transaction time is necessary when one would like to roll back the state of the database to a previous point in the time. [7] proposed four implicit times could be taken out from valid time and transaction time:

- valid time – valid-from and valid-to
- transaction time – transaction-start and transaction-stop

Figure 1 illustrates the object or information will change when any events (respect to valid time) or transactions (respect to transaction time) occurred. These time-elements could be inserted into database systems profile, the database tables or it could be stamped to the records. The vital aspect of this approach is to monitor the events and transactions for the associated data in a better manner. Besides, the process of record management becomes more efficient and effective as a result of no record overlapping.



Transaction time

FIGURE 1: Representing Time Element into Two-Dimensions Quantity

In various temporal research papers the theory of time-element can be divided into two categories: intervals and points [6][7][12]. If T is denoted a nonempty set of time-elements and d is denoted a function from T to R+, the set of nonnegative real numbers then:

$$\text{time-element, } t, = \begin{cases} \text{interval, if } d(t) > 0 \\ \text{point, otherwise} \end{cases}$$

According to this classification, the set of time-elements, T, may be expressed as $T = I \cup P$, where I is the set of intervals and P is the set of points. To extend the scope of temporal dimension, [10] have presented a model which allows relative temporal information e.g., "event A happened before event B and after January 01, 2009". [10] have suggested several temporal operators that could be used in describing the relative temporal information: {equal, before, after, meets, overlaps, starts, during, finishes, finished-by, contains, started-by, overlapped-by, met-by and after}.

3. CHANGE REQUEST MANAGEMENT PRACTICE

A change request represents a documented request for a change and management of the activities in a change process which is one of a core activity in a software process. Change management means managing the process of change as well as managing all artefacts of an evolving software system. However, it is well known that managing software changes are very difficult, complicated and time consuming. The subject matter of all change request management is the flow of all activities or procedures involved.

A good practice proposed by [13] that suggests several steps to be considered in change request management procedure (Figure 2). The petitioner submit a change request and the project manager evaluates to assess technical merit, potential side effects, overall impact to other components and functions and cost and time estimation. Results of the evaluation are documented as a change request report, which is used to make a final decision on the status approval by the configuration manager or the director of information technology unit. If the change request is approved by configuration manager, the project manager will prioritize the change request project. Before an implementation can be started, project manager will produce service order and deliver it to the project staff. This service order includes new systems specifications, project planning documentation and project contract. Upon completion of the change implementation, a software status report will be produced and a new version will be available.

To accomplish a quality change request management, [5] have suggested that an appropriate database management system is needed to be embedded into the change request management systems. This database is used to keep track of all the transactions within the project running. These transactions includes the recording of change request, project contract, validation of change request, change request approval, change request analysis, prioritizing, implementation and delivering of new version.

[2] have suggested a formal life cycle for a change request process. This life cycle provides a set of specific status for each executed change request. These statuses are: *update, close, reject* and *action.* Figure 3 shows a process flow and a set of status suggested by [2].

From study done by the authors, several weaknesses have been found in the current practice of software change request process model. These weaknesses are as follows:

- Non systematic in the procedure in change request process and it is difficult to recognize the valid time for each stage in the process;

- Many models do not consider the aspect of relative temporal in representing the valid time for each stage in change request process; and
- Most of the existing models maintain only the concept of "current view" of which an existing change request version will be overwritten when the status or information regarding to change request is updated.

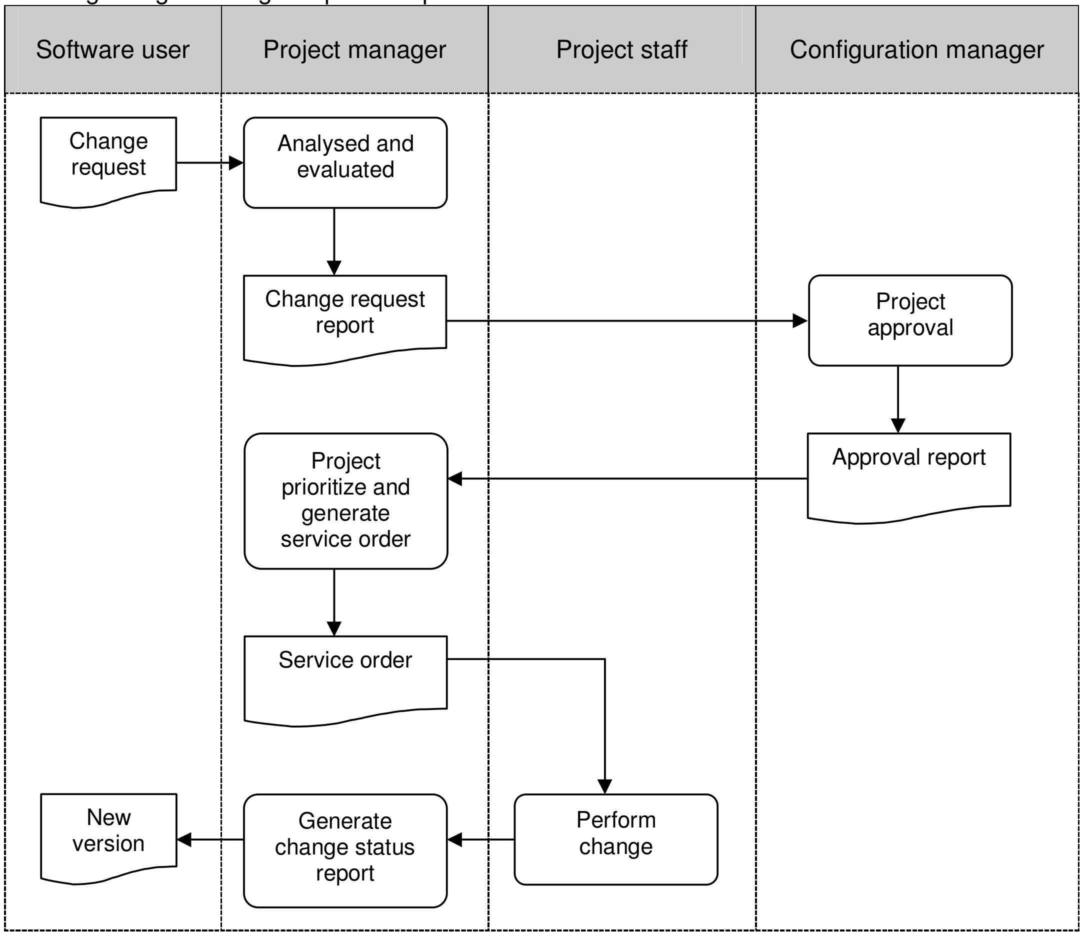


FIGURE 2: A change request procedure

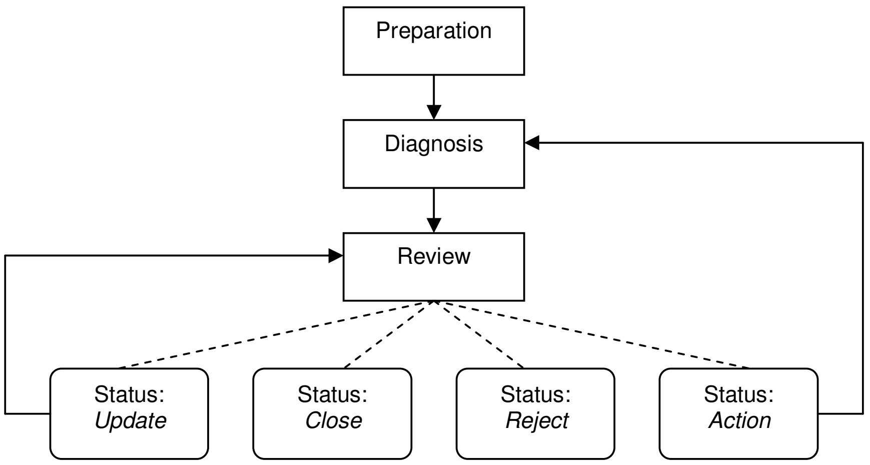


FIGURE 3: Change request life cycle

4. TEMPORAL BASED CHANGE REQUEST MANAGEMENT

Change request process is one of the main tasks in software configuration management. Updating the document of change request activities is critical to ensure valid and up-to-date information. For any change request in a software life cycle would have its own valid time and a transaction time for each activity. During unplanned changes, it is easy to forget to make updates because of the frantic nature of emergencies. Therefore, the collection of change request should be organized into systematic way for the purpose of retrieval efficiency and valid time recognition. To achieve this goal we strongly believe that, it is very important that the method of time-stamping needs to be embedded into the change request management database.

There are two main functions involved in this proposed model; *update the change request essential information* and *update the temporal information of the change request*. For each change request would have final status and can be categorized into three {In-progress, Completed and Rejected}. Furthermore, the change request would also have three values of priority {High, Moderate, Low}. There are eight change request activities considered and can be denoted as {record, contract, validation, approval, analysis, priority, implementation, deliver}. For each activity in a change request process would have it status and can be classified into two {In-progress, Completed}.

4.1 The Model of Temporal Attributes

Temporal elements such as transaction time (tt) and valid time (vt) are the main attributes to be considered in this model. Time element unit considered is in the format of [day/month/year]. Generally, in this model transaction time and valid time could be defined as:

$tt_{(i)} = \{t_i \in P\}$
$vt_{(i)} = \{t_i \in P\}$, if valid time is in point and
$vt_{(i)} = \{\tau_i \in I\}$, if valid time is in interval

where, P = a defined set of point,
I = a set of interval,
t_i = a time point at i and
τ_i = a time interval at i

Furthermore, this model also considers some temporal operators to be combined with transaction time and valid time. Among the temporal operators used in this model are equal, before, after, meets and met_by. Generally Table 1 shows the definition for each temporal operator base on a time point and a time interval. Meanwhile, figure 4 illustrates the state of temporal operators based on a define time point, t_i, and figure 5 shows the state of temporal operators based on a defined time interval, τ_i.

Temporal Operator	Time Point	Time Interval
equal	$t = \{(t = t_i) \in T\}$	$\tau = \{(\tau = \tau_i) \in T\}$
before	$\tau = \{(\tau < t_i) \in T\}$	$\tau = \{(\tau < \tau_i) \in T\}$
after	$\tau = \{(\tau > t_i) \in T\}$	$\tau = \{(\tau > \tau_i) \in T\}$
meets	$\tau = \{(\tau \leq t_i) \in T\}$	$\tau = \{(\tau \leq \tau_i) \in T\}$
met_by	$\tau = \{(\tau \geq t_i) \in T\}$	$\tau = \{(\tau \geq \tau_i) \in T\}$

TABLE 1: The definitions of temporal operator base on time point and time interval

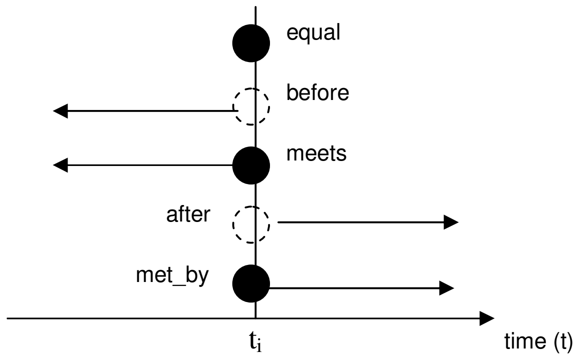


FIGURE 4: The state of temporal operators based on a time point

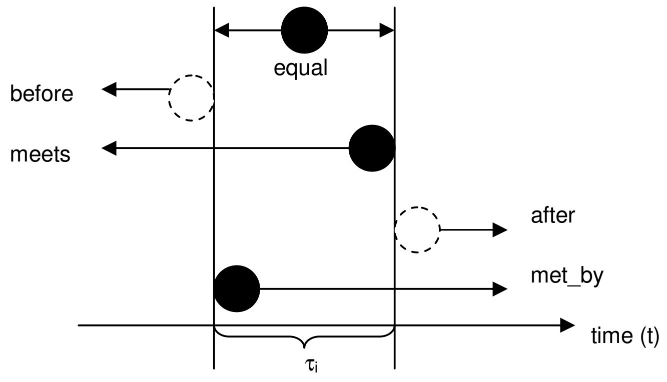


FIGURE 5: The state of temporal operators based on a time interval

4.2 Temporal Aspects in Change Request Management

As mentioned earlier, temporal elements involved here are transaction time (tt) and valid time (vt) which can be denoted, TE = {tt, vt}. Transaction time represents the date when change request status activity is recorded into the Change Request Database Management System (CRDMS). Meanwhile, valid time represents the change request time-span e.g. the date of change request report is submitted to the date of new version is required. Hence, valid time would be categorizes into two different attributes known as valid-from and valid-until. In this model, only five temporal operators will be considered, and can be denoted as OP = {equal, before, after, meets, met_by}. Therefore, if we have a set of change request signed as, C = $\{c_1, c_2, c_3, \ldots, c_n\}$ then the model is,

$$\text{TEMPORAL}(c_i \in C) \subseteq (tt \cap vt)$$

where, vt = [vt-from, vt-until]

If a change request which has a set of feature attributes A_i then a complete scheme for a temporal based in change request management can be signed as:

$$S = \{A_1, A_2, A_3, \ldots, A_n, tt, op1, vt\text{-from}, op2, vt\text{-until}\}$$

where, A_i = attribute name,
$tt \in P$,
op1, op2 $\in$ OP and
vt-from, vt-until $\in$ T

Figure 6 shows in general the insertion point of valid time and transaction time into all stages in software change request management process. As an example, we may illustrate the record transaction representing C_REQ0120's change request life cycle as in Table 2. Regarding to this model, the following convention can be used to retrieve for any temporal attribute in proposed temporal based software change request management:

- Attribute name only $\Rightarrow$ current time
- Attribute name followed by ALL $\Rightarrow$ all history
- Attribute name followed by a time point or interval $\Rightarrow$ specific time period
- Attribute name followed by RESTRICT $\Rightarrow$ specific time period designated by the condition

c_req#	tt	activity	op1	vt-from	op2	vt-until
CREQ0120	tt1	Record	op11	vf1	op21	vu1
CREQ0120	tt2	Contract	op11	vf1	op21	vu1
CREQ0120	tt3	Validation	op11	vf1	op21	vu1
CREQ0120	tt4	Approval	op11	vf1	op21	vu1
CREQ0120	tt5	Priority	op11	vf1	op21	vu1
CREQ0120	tt6	Priority	op12	vf2	op21	vu1
CREQ0120	tt7	Implementation	op12	vf2	op21	vu1
CREQ0120	tt8	Implementation	op12	vf2	op21	vu2
CREQ0120	tt9	Implementation	op13	vf2	op22	vu3
CREQ0120	tt10	Deliver	op13	vf2	op22	vu3

TABLE 2: Record transaction for CREQ0120 change request

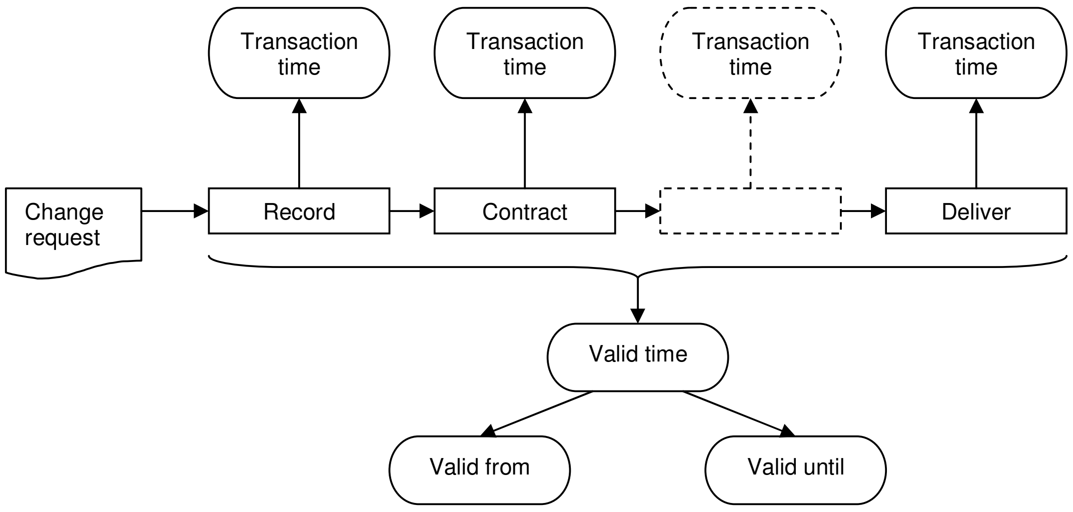


FIGURE 6: Temporal aspects in change request management

4.3 Model Evaluation

The authors have performed an evaluation process for the developed prototype of the model. In this evaluation process, 18 users (software engineers / programmers) that involved in software configuration management process are selected. The evaluation is based on the problems that constantly faced by the users. Table 3 tabulates the comments from the users regarding to the use of the model prototype in order to solve their problems.

Problem	Disagree (%)	Agree (%)	Strongly agree (%)
To validate the current status of each stage in change request process.	28	55	17
To track the history information of a maintenance process	17	72	11
No CASE tool in managing and monitoring software maintenance process	6	83	11

TABLE 3: The users perception in using the model prototype for solving their problems

5. CONCLUSION

This paper introduces a new model in change request management based on temporal elements. Here, an important issue discussed is temporal aspects such as valid time and transaction time which have been stamped on each activity in change request so that the monitoring and conflict management processes can be easily made. The proposed model also suggested that each activity should be associated with a defined progress status. We hope that the material and model presented in this paper will be useful to support future work on. For further improvements, currently, we are investigating to consider more temporal operators and developing a standard temporal model for all configuration items in software configuration managements.

6. REFERENCES

1. C. E. Dyreson, W. S. Evans, H. Lin and R. T. Snodgrass, *"Efficiently Supporting Temporal Granularities"*, IEEE Transaction on Knowledge and Data Engineering, Vol. 12 (4), 568 – 587, 2000.

2. C. Mazza, J. Fairclough, B. Melton, D. DePablo, A. Scheffer, R. Stevens, M. Jones and G. Alvisi. *"Software Engineering Guides"*. Prentice Hall. 1996.

3. D. Gao, C. S. Jensen, R. T. Snodgrass and M. D. Soo, *"Join Operations in Temporal Databases"*, The Very Large Database Journal, Vol. 14, 2 – 29, 2005.

4. S. Dart. *"Concepts in Configuration Management Systems"*. ACM – 3rd International Workshop on Software Configuration Management, 1 – 18, 1991.

5. A. Deraman & P. J. Layzell. *"A Computer-aided Software Maintenance Process Model"*. Heuristic – Journal of Knowledge Engineering. 6(1):36 – 42, 1993.

6. H. Gregerson & C.S Jensen. *"Temporal Entity-Relationship Models – A Survey"*. IEEE Transaction on Knowledge and Data Engineering. 11(3): 464 – 497, 1999.

7. C. S. Jensen & R.T. Snodgrass. *"Temporal Data Management"*. IEEE Transaction on Knowledge and Data Enginering. 11(1):36 – 44, 1999.

8. G. Joeris. *"Change Management Needs Integrated Process and Configuration Management"*. 6th European Software Engineering Conference, 125 – 141, 1997.

9. B. Knight & J. Ma. *"A General Temporal Theory"*. The Computer Journal. 37(2):114-123, 1994.

10. B. Knight & J. Ma. *"A Temporal database Model Supporting Relative and Absolute Time"*. The Computer Journal. 37(7):588 – 597, 1994.

11. K. Torp, C. S. Jensen and R. T. Snodgrass, *"Effective Timestamping in Database"*, The Very Large Database Journal, Vol. 8, 267 – 288, 1999.

12. D. H. O. Ling & D. A. Bell. *"Modelling and Managing Time in Database Systems"*. The Computer Journal. 35(4):332 – 342, 1992.

13. R. F. Pacheco. *"Keeping the Software Documentation Up to Date in Small Companies"*. CLEI Electronic Journal. 3(1), 2000.

Component Selection Efforts Estimation– a Fuzzy Logic Based Approach

Kirti Seth kirti.twins@gmail.com
Lecturer, CSE/GCET
UPTU
Greater Noida,India

Arun Sharma
Professor, AIIT arunsharma2303@gmail.com
Amity University,
Noida,India

Ashish Seth aseth@amity.edu
Lecturer, AIIT
Amity University
Noida, India

Abstract

Effort Estimation with good accuracy helps in managing overall budgeting and planning. The accuracy of these estimates is very less and most difficult to obtain, because no or very little detail about the project is known at the beginning. Due to architectural difference in CBS (Component Based Systems), this estimation becomes more crucial. Component-based development mainly involves the reuse of already developed components. The selection of best quality component is of prime concern for developing an overall quality product. CBS mainly involves two types of efforts: selection and integration. Present paper presents a fuzzy rule based model for estimating the efforts in selecting these Components for developing an application using CBSE approach.

Keywords: Component, Component Based software Engineering, Selection Efforts, Reusability, Fuzzy Rule Base.

1. INTRODUCTION

Commercial off the Shelf (COTS) software components or Component Based Software come from different source and have varied characteristics but by integrating them a software system can be formed [2]. Component Based Software Development (CBSD) approach is used widely in software industry. Efficient application can be developed through the integration of these components by using this approach. The components selected for the application are behind the success of these applications. Therefore a large amount of efforts has to be invested for selecting these components. There are several approaches for estimating such efforts. Present work proposes a fuzzy logic based approach for component selection. We identified the main factors which are useful in taking decision for selecting best suitable component. These factors are Reusability, Portability, Functionality, Security, and Performance. A rule base is prepared based on the effects of these on overall selection of component.

2. SELECTION EFFORTS

The main activities in any component based development require the following sequence of activities for the components.

Search -> Select -> Create/Adapt -> Integrate -> Maintain

Search and selection are the two most important activities, on which the quality of entire application depends. Therefore, sufficient efforts must be invested on selection of appropriate and better quality component. When a component is selected among various available components, a large amount of efforts is invested. These efforts should be estimated for better budgeting and planning of the software development. A lot of research work has been conducted by the researchers. Ali et al. [11] proposed the use of fuzzy sets in COCOMO 81 model [12]. They measured each cost driver of the intermediate COCOMO'81 model on a scale of six linguistic values ranging from very low, to nominal to extra high. For each cost driver and its associated linguistic values, they defined the corresponding fuzzy sets. These fuzzy sets are represented by trapezoidal shaped membership functions to finally estimate the overall cost. Musilek et al. [13] proposed F-COCOMO model, using fuzzy sets. F-COCOMO was based on fuzzy sets. Fuzzy sets may be applied to other models of cost estimation such as function point method. Mattson et al. [15] found that the software cost estimation model may be more user friendly by using concept of fuzzy sets. Valerie Maxville et al. [9] give a context driven component evaluation approach for estimating Component Selection Efforts. Mustafa Nouri et al. [14] estimate component selection by considering the NP-complete process of selecting a minimal set of components to satisfy a set of objectives. For this process, authors designed three variations of component selection and used approximation theory to find near optimal solution.

3. PROPOSED APPROACH

In this approach, following five factors that affect the Component Selection Efforts have been proposed.

3.1 Reusability

In case of Component Based Development, software reuse refers to the utilization of a software component with in an application, where the original motivation for constructing this component was other than for use in that application. In other words, reuse is the process of adapting a generalized component to various contexts of use. The idea of reusing software embodies several advantages. It improves productivity, maintainability and quality of software. Reusability is among one of the most basic properties of Component Based Development (CBD). In this model reusability is used as a factor. It is considered that if there are several components of the same type and among those available components the most appropriate component will be the one that has been reused so many times. So to select this component efforts invested will be low. Hence the selection efforts are indirectly proportional to the reusability.

3.2 Portability

This factor is defined as the ability of a component to be transferred from on environment to another with little modifications, if required. The component should be easily and quickly portable to specified new environments if and when necessary, with minimized porting costs and schedules. Therefore the specification of the component should be platform independent. Some components are platform independent that are highly portable. If a component is easily portable then the selection efforts are low.

3.3 Functionality

Functionality of a component depends upon the number of functions and their properties in these functions. It means that the component should provide the functions and services as per the requirement when used under the specified condition. Pre existing components with or minimum changes will allow low cost, faster delivery of end product. If the functionality is high then the efforts invested are also high.

3.4 Security

The primary goals of software security are the preservation of the confidentiality, integrity, and availability of the information assets and resources that the software creates, stores, processes, or transmit, including the executing programs themselves. In other words, users of secure software have a reasonable expectation that their data is protected from unauthorized access or modification and that their data and applications remain available and stable. It refers how the component is able to control the unauthorized access to its provided services [5].If the component is highly secure then the efforts invested in selecting that component will be low.

3.5 Performance

This characteristic expresses the ability of a component to provide appropriate performance. This is affected by the component technology, mainly through resource usage by the run-time system but also by interaction mechanism. Component can be internally optimized to improve performance without affecting their specifications. Component should be tested on various platforms to check the performance. It means how well a component is providing the results. That is if all the parameters are given values then how accurately and fast it can produces the results [5]. Component Selection efforts are indirectly proportional to the performance.

4. FUZZY LOGIC

Fuzzy logic was proposed by Zadeh in 1965 [8] and since then it has been the subject of important investigations. It is a mathematical tool for dealing with uncertainty and also it provides a technique to deal with imprecision and information granularity. A keen mapping between input and output spaces may be developed with the help of fuzzy logic .Some major modules of fuzzy logic are as follows:

First stage transformed the classification tables into a continuous classification, this process is called fuzzification. These are then process in fuzzy domain by inference engine based on knowledge base (rule base and data base) supplied by domain experts. Finally the process of translating back fuzzy numbers into single “real world” values is named defuzzification.

5. PROPOSED FUZZY MODEL

There are five inputs to this fuzzy model, namely Reusability, Portability, Functionality, Security, and Performance. Figure 1 shows the fuzzy model.

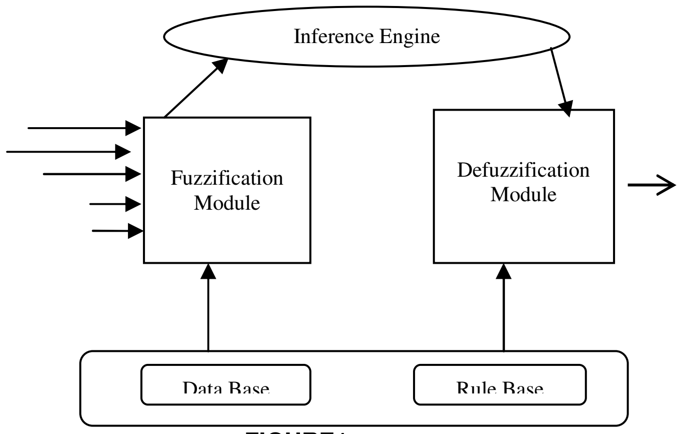


FIGURE1. Fuzzy Model.

This model considers all five inputs and provides a crisp value of Selection efforts using the Rule Base. All inputs can be classified into fuzzy sets viz. Low, Medium and High. The output Selection Efforts is classified as Very High, High, Medium, Low, and Very Low. In order to fuzzify the inputs, the following membership functions are chosen namely Low, Medium High. They are shown in Fig 2.

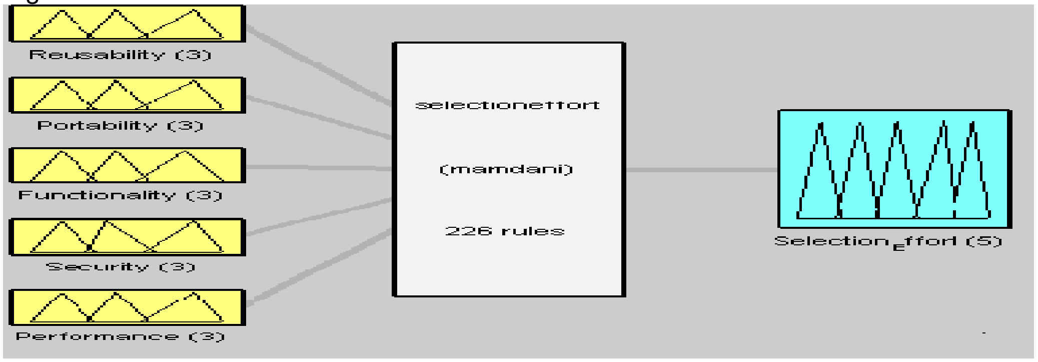


FIGURE2 Inputs and Outputs in our Fuzzy Model

Similarly the output variable i.e. Selection effort has five membership functions. All the inputs and outputs are fuzzified as shown in figure 2. All possible combination of inputs were considered which leads to 3^5 i.e. 243 sets. Selection Effort in case of all 243 combinations is classified as Very High, High, Medium, Low, very low by expert opinion. These lead to the Formation of 243 rules for the fuzzy model and some of them are shown below:

- If (Reusability is low) and (Portability is low) and (Functionality is low) and (Security is low) and (Performance is low) then Selection Efforts are very high. Selection Efforts are very high.
- If (Reusability is low) and (Portability is low) and (Functionality is low) and (Security is medium) and (Performance is low) then Selection Efforts are high.
- If (Reusability is low) and (Portability is low) and (Functionality is medium) and (Security is low) and (Performance is low) then Selection Efforts are very high formation of 243 rules for the fuzzy model and some of them are shown below:
- If (Reusability is low) and (Portability is low) and (Functionality is low) and (Security is low) and (Performance is low) then Selection Efforts are very high. Selection Efforts are very high.
- If (Reusability is low) and (Portability is low) and (Functionality is low) and (Security is medium) and (Performance is low) then Selection Efforts are high.
- If (Reusability is low) and (Portability is low) and (Functionality is medium) and (Security is low) and (Performance is low) then Selection Efforts are very high.

….
….

All 243 rules are entered and a Rule Base is created. a rule will be fired depending on a particular set of inputs. Mamdani style of inference is used.

System	Name='Selectioneffort', Type='mamdani', Version=2.0, NumInputs=5, NumOutputs=1, NumRules=243, AndMethod='min', OrMethod='max', ImpMethod='min', AggMethod='max', DefuzzMethod='centroid'
Input1	Name='Reusability', Range=[0 1], NumMFs=3, MF1='Low':'trimf',[0 0.16 0.33], MF2='medium':'trimf',[03 0.45 0.62], MF3='high':'trimf',[0.57 0.85 1]
Input2	Name='Portability', Range=[0 1], NumMFs=3, MF1='low':'trimf',[0 0.16 0.34], MF2='medium':'trimf',[0.30 0.45 0.62], MF3='high':'trimf',[0.56 0.85 1]

Input3	Name='Functionality', Range=[0 1], NumMFs=3, MF1='low':'trimf',[0 0.16 0.35], MF2='medium':'trimf',[0.30 0.45 0.62], MF3='high':'trimf',[0.56 0.80 1]
Input4	Name='Security', Range=[0 1], NumMFs=3, MF1='low':'trimf',[0 0.16 0.34], MF2='medium':'trimf',[0.3 0.40 0.65], MF3='high':'trimf',[0.60 0.85 1.0]
Input5	Name='Performance', Range=[0 1], NumMFs=3, MF1='low':'trimf',[0 0.16 0.33], MF2='medium':'trimf',[0.3 0.45 0.62], MF3='high':'trimf',[0.58 0.85 1.0]
Output1	Name='Selctioneffort', Range=[0 1], NumMFs=5, MF1='Very_Low':'trimf',[0 0.12 0.23], MF2='Low':'trimf',[0.2 0.32 0.42], MF3='Medium':'trimf',[0.40 .51 0.62], MF4='High':'trimf',[0.60 0.75 0.82], MF5='Very_High':'trimf',[0.80 .91 1.0]

Selection efforts are observed for a particular set of inputs using MATLAB Fuzzy tool box [8].

Table 1: Inputs and Outputs for Fuzzification

6. EVALUATION OF THE PROPOSED MODEL

We use the proposed model for five components and estimated the cost for each of these components. These selection efforts are on a scale of 0 to 1.

Project	Reusability	Portability	Functionality	Security	Performance	Selection efforts
P1	0.50	0.50	0.50	0.50	0.50	0.51
P2	0.17	0.45	0.23	0.62	0.18	0.472
P3	0.12	0.20	0.09	0.13	0.10	0.902
P4	0.72	0.90	0.75	0.85	0.97	0.311
P5	0.58	0.85	0.62	0.32	0.90	0.541

Table2: Results using rule Based system

From the table, it is clear that component P4 has the least selection effort i.e. will result in low cost development of the end product, while if the component P3 is to be used, the selection efforts were very high. So we can say that our model is able to predict the effort invested in selecting a component.

7. CONCLUSION:

For component-based development, efforts are mainly invested in selecting the appropriate component and then integrating it in the application. In this paper we have proposed a fuzzy rule based approach for estimating component selection efforts for these systems. The proposed approach is used to estimate efforts on some real time projects. However, the work further requires validation. For this purpose we are collecting more data from projects and by using Analytical Hierarchical approach we will validate our results.

8. REFERENCES

[1] Capers Jones, "Software Estimating Rules of Thumb", available at http://www.ieeexplore.ieee.org/iel1/2/10412/ 00485905.pdf.

[2] Richard E. Fairley "Recent Advances in Software Estimation Techniques", Management Associates Woodland Park, CO, USA 1992.

[3] I. Sommerville, "Software Engineering", Sixth Edition", Addison-Wesley Publishers Limited, 2001.

[4] IEEE standard Glossary of Software Engineering Technology, IEEE Std. 610 12-1990.

[5] A. Sharma, R. Kumar, P. S. Grover, "Empirical Evaluation and Validation of Complexity Metrics for Software Components", International Journal of Software Engineering and Knowledge Engineering, Vol. 18, Issue 7, 2008, pp: 919-931.

[6] P. S. Grover, R. Kumar, A. Sharma, "Few Useful Considerations for Maintaining Software Components and Component Based Systems", ACM SIGSOFT Software Engineering Notes, Vol. 32, Issue 5, 2007, pp: 1-5.

[7] Voas Jeffrey, Agresti, William W., Software Quality from a Behavioral Perspective IEEE Computer Society IT Pro, July- August 2004.

[8] Roger Jang and Ned Gulley, Fuzzy Logic Toolbox for MATLAB. User's Guide. The Math Works Inc, USA, 1995.

[9] Valerie Maxville, Jocelyn Armarego, Cheiou Peng Lam, "Intelligent Component Selection", proceedings of 28th annual international computer society and application conference.(COMPSAC '04).

[10] Carr R.D., Doddi, S., Konjevod, G., Marathe, and M.V.: On the red-blue set cover problem. In: SODA, 2000, pp. 345-353.

[11] Ali, I., A., Alain and K. Laila, "COCOMO cost model using Fuzzy Logic", 7th International Conference on Fuzzy Logic and Technology, Atlantic, New Jersey, March-2000.

[12] Bohem, B., "Software Engineering Economics", 1st Edition, Prentice-hall, Englewood Cliffs, New Jersey, 1981, ISBN: 0138221227.

[13] Musilek, P., W. Pedrycz, G. Succi and M. Reformat, "Software cost estimation with fuzzy models" ACM SIGAPP Applied Computer Review, Vol. 8, pp:24-29.

[14] Mostafa Nouri and Jafar Habibi, 2008, "Approximating Component Selection with General Costs"CSICC 2008, CCIS 6, pp: 61-68, .

[15] Matson, J.E., B.E. Barrett and J. M. Mellichamp, "Software Development Cost Estimation using Function Points", IEEE Transactions Software Engineering, Vol. 20, pp: 275-287.

Discovery of Frequent Itemsets Based on Minimum Quantity and Support

Preetham Kumar preetham.kumar@manipal.edu
Senior Lecturer
Department of Information and
Communication Technology
Manipal Institute of Technology
Manipal University
Manipal, 576104, India

Ananthanarayana V S anvs@nitk.ac.in
Professor & Head
Department of Information Technology
National Institute of Technology Karnataka,
Surathkal, 575025, India

Abstract

Most of the association rules mining algorithms to discover frequent itemsets do not consider the components of transactions such as total cost, number of items or quantity in which items bought. In a large database it is possible that even if the itemset appears in a very few transactions, it may be purchased in a large quantity for every transaction in which it is present, and may lead to a very high profit. Therefore the quantity and the total cost of the item bought are the most important components, lack of which may lead to loss of information. Our novel method discovers all frequent itemsets based on items quantity, in addition to the discovery of frequent itemsets based on user defined minimum support. In order to achieve this, we first construct a tree containing the quantities of the items bought, as well as the transactions which do not contain these items in a single scan of the database. Then, by scanning the tree, we can discover all frequent itemsets based on user defined minimum quantity as well as support. This method is also found to be more efficient than the Apriori and the FP-tree, which require multiple scans of the database to discover all frequent itemsets based on user defined minimum support.

Keywords: Confidence, Minimum total cost, Number of items, Quantity, Support.

1. INTRODUCTION

Data mining is the extraction of the hidden predictive information from large databases. It is a powerful new technology with great potential to analyze important information stored in large volumes of data. It is one of the steps in knowledge discovery in databases. The goal of knowledge discovery is to utilize the existing data to find new facts and to uncover new relationships that were previously unknown, in an efficient manner with minimum utilization of space and time. There are several data mining techniques [2]. One of them is Association Rules Mining. Among the areas of data mining, the problem of deriving association [1, 2] from data has received a great deal of attention. This describes potential relations among data items (attribute, variant) in databases. Agarwal et al [1] formulated the problem in 1993. In this problem, we are given a set of items and a large collection of transactions, which are subsets of these items. The task is to find relationship between the presences of various items within this database. An item is a thing that is sold in each transaction. For every item, its item number appears in a particular transaction. Some transactions also contain relevant information of the transactions such as, quantity in which it is bought, cost of the item, customer's age, salary etc. An itemset [1] (this term is used for item set in all books and papers on data mining) is a set of all items. Let $A = \{I_1, I_2, \dots I_m\}$ be a set of items. Let D, the transaction database, be a set of transactions, where each transactions t is a set of items. Thus, t is a subset of A. A transaction t is said to support an item I_i , if I_i, is present in t. Further, t is said to support a subset of items X contained in A, if t supports each item in X. An itemset X contained in A has a support s in D, if s% of transactions in D support X. For a given transaction database D, an association rule[1] is an expression of the form X->Y, where X and Y are the sets of A and where $X \subseteq A$, $Y \subseteq A$, and $X \cap Y = \Phi$. The intuitive meaning of such a rule is that the transaction of the database which contains X tends to contain Y. The rule X->Y has support s in a given transaction database D if s% of transactions in D support XUY (i.e., both X and Y). This is taken to be the probability, $P(X \cap Y)$. The rule X->Y has confidence c in the transaction database D if c percentage of transactions in D containing X that also contain Y. This is taken to be the conditional probability, $P(Y|X)$. That is, Support(X->Y)=$P(X \cap Y)$=s, Confidence(X->Y)=$P(Y|X)$ = Support(X->Y)/Support (X)=c.

Let D be the transaction database and s be the user specified minimum support. An itemset X contained in A is said to be frequent in D with respect to s, if support value of X is greater than or equal to s. Every frequent itemset satisfies downward closure property, i.e every subset of frequent itemset is frequent.

Mining of association rules is to find all association rules that have support and confidence greater than or equal to the user-specified minimum support and minimum confidence respectively [2,3,4,6]. This problem can be decomposed into the following sub problems:

a) All itemsets that have support above the user specified minimum support are discovered. These itemset are called frequent itemsets.

b) For each frequent itemset, all the rules that have user defined minimum confidence are obtained.

There are many interesting algorithms proposed recently and some of the important ones are the candidate generation based Apriori and its variations and non candidate generation based algorithms such as FP-tree, PC- Tree algorithms.

The algorithm Apriori called as level-wise algorithm operates in a bottom-up, breadth-first search method. It is the most popular algorithm to find all frequent sets, proposed in 1994 by Agrawal et al [7]. It makes use of the downward closure property. The nicety of the method is that before reading the database at every level, it graciously prunes many of the sets, which are unlikely to be frequent.

The number of database passes is equal to the largest size of the frequent itemset. When any one of the frequent itemsets becomes longer, the algorithm has to go through several iterations and, as a result, the performance decreases.

The FP- Tree Growth algorithm is proposed by Han et al [2, 3]. It is a non candidate generation algorithm and adopts a divide and conquers strategy. The following steps are used. (i) Compress the database representing frequent items into FP-tree, but retain the itemset association information (ii) Divide such a compressed database into a set of conditional databases, each associated with one frequent item (iii) Mine each such database separately.

This algorithm requires two scans of the database to discover all frequent sets. The main idea of the algorithm is to maintain a Frequent Pattern Tree of the database.

A frequent pattern tree is a tree structure consisting of an item-prefix-tree and a frequent-item-header table. The FP-Tree is dependent on the support threshold. For different values of threshold the trees are different. Also, it depends on the ordering of the items. The ordering that is followed is the decreasing order of the support counts.

When the database is large, it is sometimes unrealistic to construct a main memory based FP-tree. An interesting alternative is to first partition the database into a set of projected databases,

and then construct an FP-tree and mine it in each projected databases. Such a process can be recursively applied to any projected databases if its FP-tree still can not fit in main memory.
A study on the performance of the FP-growth method shows that it is efficient and scalable for mining both long and short frequent patterns, and is about an order of magnitude faster than the Apriori algorithm.
The Pattern Count tree (PC-tree) is the contribution of V. S. Ananthanarayana et al [5] which is a complete and compact representation of the database. PC-tree is a data structure, which is used to store all the patterns occurring in the tuples of a transaction database, where a count field is associated with each item in every pattern, which is responsible for compact realization of the database.
Each node of the PC-tree consists of (i) item-name (ii) count and (iii) two pointers called child pointer(c) and sibling pointer(s).
In the node, the item-name field specifies the item that the node represents, the count field specifies the number of transactions represented by a portion of the path reaching this node, the c-pointer field represents the pointer to the following pattern and the s-pointer field points to the node which indicates the subsequent other patterns from the node under consideration.
It is proved that, construction of PC-tree and generation of all large itemsets requires a single database scan. Because of compactness of PC-tree, the algorithms based on PC-tree are scalable. Also, in ordered to discover large itemsets a unique ordered FP-tree, called Lexicographically Ordered FP-tree is constructed from a PC-tree without scanning the database.

1.1 Motivation

One of the key features of most of the existing algorithms is that they assume underlying database size is enormous, and involves either a candidate generation process or a non-candidate generation process. The algorithms with candidate generation process require multiple passes over the database and are not storage efficient. In addition, the existing algorithms discover all frequent itemsets based on user defined minimum support without considering the components such as quantity, cost and other attributes which lead to profit.
Consider for example a sample database given in Table 1 in which every element of each transaction represents either the quantity of the respective attribute or the item.

TABLE 1: Sample Database

TID/Attributes	A	B	C	D
1	10	5	0	0
2	0	0	3	0
3	4	0	4	0
4	5	2	5	0
5	0	0	0	10

If an itemset appears in a very few transactions but in a large quantity, then it is possible that buying of this itemsets leads to profit, will not qualify as frequent itemset based on user defined minimum support. This results in a loss of information. In the sample database given in Table 1, if the user defined minimum support is 2 transactions, then an item D is not frequent and will not appear in the set of frequent itemsets, even though it is bought in a large quantity and leads to more profit than other frequent items. This motivated us to propose the following method which discovers all frequent itemsets based on user defined minimum quantity. With this method it is also possible to discover frequent itemsets based on user defined minimum support.

1.2 Frequent itemset and quantity based Frequent itemset or weighted minimum support

If an itemset satisfies user defined minimum support then we call it a frequent itemset. If an itemset satisfies user defined minimum quantity then we say that it is a quantity based frequent itemsets. The weight of an itemset is the ratio of the quantity in which the itemset is bought to the number of transactions in which it is present. For example, if a user defined minimum quantity is equal to 4 then the items A, C, D become quantity based frequent one itemets.

2. PROPOSED METHOD

Our novel method uses the concept of tree called Q-TID Tree. The Q-TID tree consists of information regarding items, the quantity in which the items have been purchased and the TIDs in which these items are not present**.** The structure of this tree is as follows.

2.1 Structure of the Q-TID

The Q-TID tree has three different nodes and is shown in Figure 1.

(i) The first type of node is labeled with item name or attribute name and two pointers, one pointing to the nodes containing quantity, and another is a child pointer pointing to the next attribute. This node represents the head of that particular branch.

(ii) The second type of node labeled as Quantity1, Quantity2 etc, indicates which particular item is purchased. This node has only one pointer pointing to the next object having this particular attribute.

(iii) The third type of node, appears as the last node in every branch of the tree, and is similar to the second type, but consists of information corresponding to the transactions ids(TIDs), which do not contain the particular item. This information represents a whole number which is obtained, forming prime product of prime numbers. If this product is factorized, then it is possible to obtain all the TIDs which do not contain the particular item.

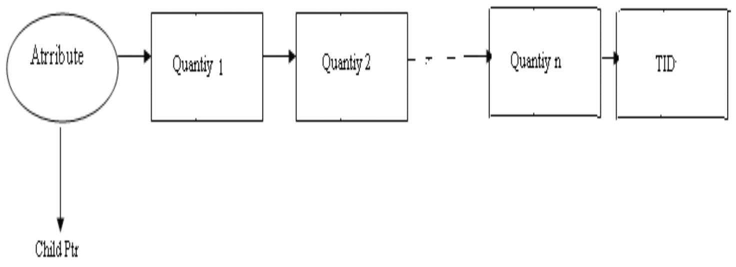


FIGURE 1: A Branch of Q-TID Tree

2.2 Forming the product corresponding to the TIDs

In this step mapping of every transaction which does not contain the particular item to a prime number, and then its product, is obtained with the already existing number in the last node of every branch of the tree. The prime numbers assigned to the TID are shown in Table 2. For example, the first prime number 2 is assigned to 1st TID and a second prime number 3 is assigned to the 2nd TID and so on.

TABLE 2: Prime Number Table

Positive integers	1	2	3	4	5	6	7	8	9
Prime numbers	2	3	5	7	11	13	17	19	23

The prime numbers are used to make our process very simple because of their following property.
If a, b are any two prime numbers with multiplicity m and n then their product is $a^m b^n$. With this product, it is possible to obtain the original prime numbers used for forming the product.
For example, consider the prime numbers 2, 3, 5 and its product is 30. With this product it is possible derive all prime numbers involved in forming the product. Therefore one can see that the prime numbers 2, 3 and 5 are involved in forming the product. In our case, these numbers represent the TIDs 1, 2 and 3. The Q-TID tree corresponding to the sample database given in Table 1 is shown in Figure 2.

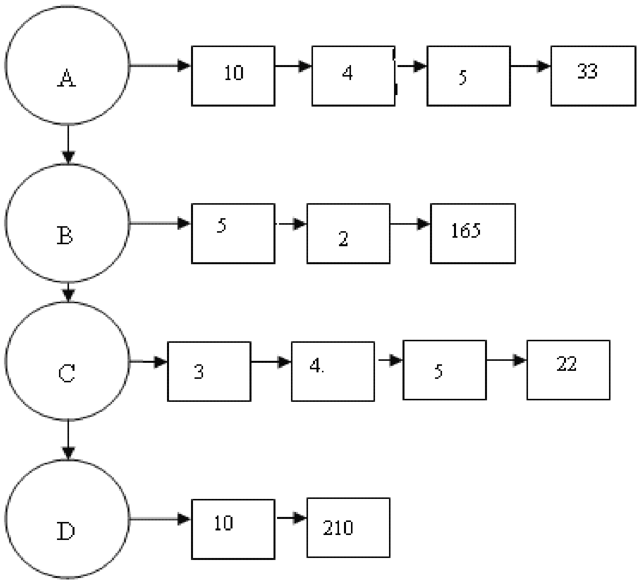


FIGURE 2: Q-TID tree for Table1

The Q-TID tree algorithm involves following steps. They are

A Construction of Q-TID Tree.
B Removal of infrequent items based on user defined minimum quantity of Q-TID Tree
C Discovery of frequent itemsets based on user defined minimum quantity
D Discovery of frequent itemsets based on user defined weighted minimum support

A Algorithm for constructing Q-TID Tree

```
Input:    The database D
Output:  Q-TID tree
Create an attribute or item node labeled with respective item and its corresponding TID
node labeled with 1
for each item I in a transaction t € D
do begin
            create a node labeled with its quantity and add
            these nodes to the respective item or attribute
            node.
            If I is not in t
            find t^th prime number and its product with
            already existing prime number in the last node TID of
            the branch corresponding to I.
end
```

B Algorithm for Reducing the Q-TID Tree

Input : weighted_min_qsup =user specified minimum quantity

```
        n = number of transactions in which a particular item is present
    Output: Q-TID tree contains only frequent items based on
        user defined quantity in addition to the information corresponding to the TIDs which
        do not contain a particular item.
    for each item or attribute in a Q-TID tree
    do begin
            If sum (quantities of all nodes except a last node/n) < weighted_min_qsup then
            remove those nodes corresponding to the quantities from the branch which
            originates from the attribute node labeled with I
    end
```

C Algorithm to discover all frequent itemsets based on quantity

```
Input: A Reduced Q-TID tree, weighted_min_qsup=user specified minimum quantity.
        F= {set of all frequent one itemsets based on quantity}
        P=set of all non empty subsets of F excluding the sets containing one attribute.
Output: set of all frequent itemsets =$F_Q$ .
begin
    $F_Q$ = { set of all frequent attributes or one itemset based on quantity }
    for each f in P do
            begin
                T = { TIDs of first attribute in f }
                    for each m in f other than first attribute do
                    begin
                            T=T ∩ { TIDs of m}
                    end
                If T is non empty then
                    If (sum of quantities of T /|T| ) >= weighted_min_qsup
                            $F_Q$ = $F_Q$ U f
            end
```

D Algorithm to discover all frequent itemsets based on minimum support.

Input: A Reduced Q-TID tree, min_sup=user specified minimum support, N =Total number of transactions.
L= {set of all frequent one itemsets based on minimum support}
P(L) ={set of all non empty subsets of L excluding the sets containing one attribute or one item }.
F_M = {set of all frequent attributes or one itemset based on minimum support}
Output: set of all frequent itemsets. F_M

```
for each f in P
do begin
    T = {prime factors corresponding to TIDs which do not contain the first attribute in f }
    for each m in f other than first attribute
    do begin
            T=T U { prime factors of corresponding TID s which do not contain m}
    end
    If T is non empty
        If ( N - | T | ) >= min_sup
            $F_M$ = $F_M$ U f
end
```

Illustration:
Consider for example, a sample database given in Table 1,
If we consider weighted_min_qsup = 4 then the attributes A, C and D will be frequent in the database. The Reduced Q-TID is shown in Figure 3.

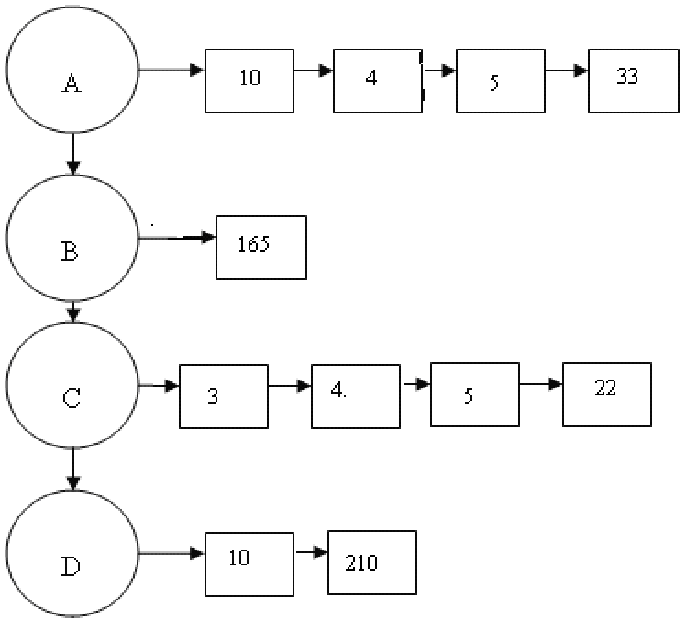


FIGURE 3: A Reduced Q-TID Tree

we observe that **F$_Q$** ={A, C, D} . Therefore
P = {{A, C}, {A, D}{C,D}, {A, C, D} }
Consider a set {A, C}, which appears in transactions 3 and 4.
i.e T={3,4}.
The TIDs are obtained by using the following procedure .
The last node of the item A corresponding to the TID contains 33, which involves prime numbers 3 and 11. Similarly, the last of C contains 22, which involves prime numbers 2 and 11. The union of these numbers will give us set {2, 3, 11}. These numbers correspond to the transaction 1, 2, 5. Hence both A and C will occur only in transactions 3 and 4. Further, the sum of their corresponding quantities is equal to 9+9=18 and weighted support of {A,C} = 18/2 =9.
Hence {A, C}is frequent.
By similar arguments, we found that the sets {A, D}, {C, D} and {A,C, D} are infrequent sets.
Hence **F$_Q$** = { {A}, {C}, {D},{ A, C}}
Now if user defined minimum support is equal to 2 then the items A, B, C will be frequent one itemsets.
Therefore **L =** {A, B, C} . Therefore
P(L) = { {A, B}, {A,C}, {B,C}, {A, B, C} }. Now applying above algorithm, we see that **F$_M$** = { {A}, {B}, {C}}.
It can be observe from the Q-TID tree that itemsets
{A, B} is not present in transactions 2, 3 and 5.
{A, C } is not present in transactions 1, 2 and 5.
{B, C} is not present in transactions1, 2, 3 and 5.
{A, B, C} is not present in transactions 1, 2, 3 and 5.
Since N =5, we have supports of {A, B} = 2, {A, C } = 2, {B, C}= 2 and {A, B, C} =1. This shows that the itemset {A, B, C } is not frequent.
Now **F$_M$** = { {A}, {B}, {C}, {A, B}, {A,C}, {B,C}}.

Note : If an item is present in every transaction, then the node corresponding to the TIDs which do not contain that particular item will not appear in the Q-TID Tree.

The following example illustrates this situation. Consider a sample database given in Table 3 which has 3 attributes and 3 tuples.

TABLE 3: Sample Database

TID	A1	A2	A3
T1	3	2	1
T2	6	2	1
T3	2	2	4

The Q-TID tree corresponding to Table 3 is given in Figure 4. It does not contain nodes corresponding to the TIDs since every item is present in every transaction.

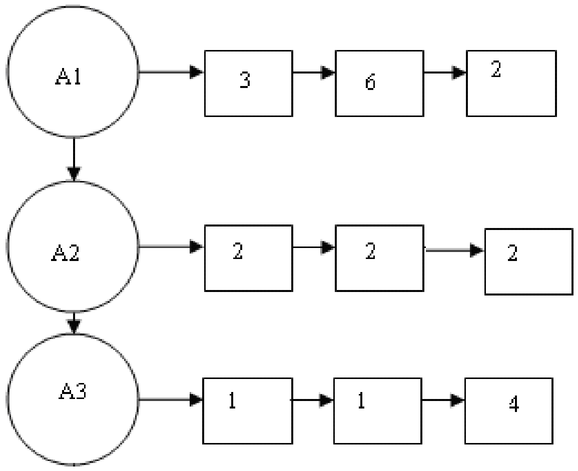


Figure 4: Q-TID Tree of Table3

3. PERFORMANCE ANALYSIS

Theoretical Analysis

The algorithm consists of four steps

A Construction of Q-TID Tree

If the given database contains N transactions then this can be done in O(N) time.

B Removal of infrequent items based on user defined weighted minimum support of Q-TID Tree

If there are m items which are not frequent then all the nodes containing quantity information of these m attributes are deleted. If there are on an average g nodes for every attribute then this step is in O(mg).

C Discovery of frequent itemsets based on user defined weighted minimum support

The major step is the process of finding power set P. If there are n frequent 1-item attribute sets then this step is in $O(2^n)$.

D Discovery of frequent itemsets based on user defined minimum support

The major step is the process of finding P(L). If there are n frequent 1-item attribute sets then this step is in $O(2^n)$

4. EXPERIMENTAL ANALYSIS

For the performance analysis, a simulation of buying patterns[10] of the customers in retail patterns was generated and in the data set which we used every attribute value of the transaction

was considered as quantity of the corresponding attribute in the database**.** We compared our algorithm with FP- tree and Apriori and found that ours was time efficient.
The above algorithm is implemented and used for discovering frequent itemsets based on quantity as well as minimum support for data sets containing the transactions 100, 500, 1000, 5000 with 20 attributes or items. The time required to discover all frequent itemsets is shown Figure 5 and Figure 6 respectively.

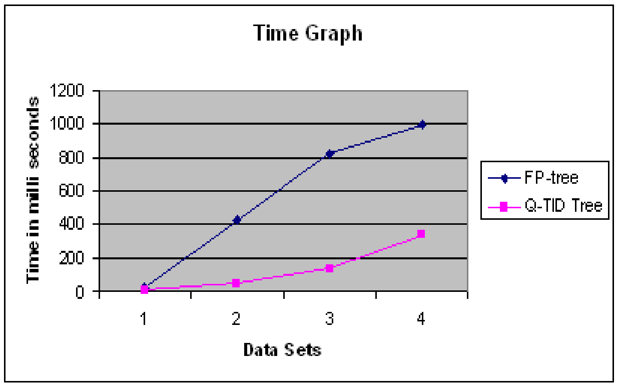


FIGURE 5: Q-TID Vs FP-Tree

The proposed algorithm is superior to the FP-tree algorithm in following ways:

1. Scans database only once.
2. No sorting of each item of the transaction.
3. No repeatedly searching the header table for maintaining links, while inserting a new node into tree.

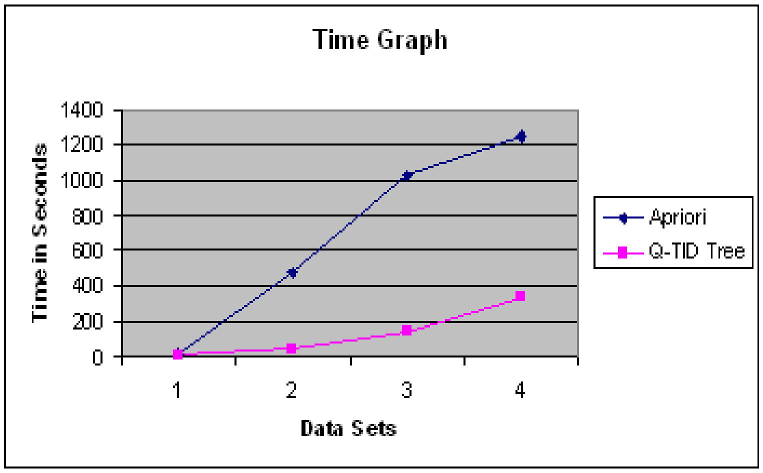


FIGURE 6: Q-TID Vs Apriori

The proposed algorithm is superior to the Apriori algorithm in the following ways:

1. Scans database only once.
2. It does not involve candidate generation method.

5. CONCLUSION

The Q-TID Tree Algorithm is a new method for finding frequent itemsets based on user defined quantity and minimum support. It is found that this algorithm is time efficient when compared to the FP-tree and Apriori. Since we have used small data sets, this algorithm can be fine tuned with large databases. This method may be modified further to store TIDs more efficiently.

6. REFERENCES

[1] Jiawei Han Micheline Kamber, Data Mining Concepts and Techniques .Morgan Kaufman, San Francisco, CA, 2001

[2] Han, J., Pei, J., Yin, Y. “Mining Frequent Patterns without Candidate Generation”, Proc. of ACM-SIGMOD International Conference Management of Data. Dallas, 2000, TX, 1-12.

[3] Han J, Pei Jian, Runying Mao ” Mining Frequent Patterns without Candidate Generation: A Frequent Pattern Tree Approach” , Data Mining and Knowledge Discovery, Kluwer Academic Publishers, Netherland” 2004, pp 53-87.

[4] R. Hemlata, A. Krishnan, C. Scenthamarai, R. Hemamalini. ”Frequent Pattern Discovery based on Co-occurrence Frequent Tree”. In Proceeding ICISIP-2005.

[5] Ananthanarayana V. S, Subramanian, D.K., Narasimha Murthy M. “Scalable, distributed and dynamic mining of association rules” In Proceedings of HIPC'00. Springer Verlag Berlin,Heidelberg,2000, 559-566.

[6] R. Srikant and R. Agarwal. “Mining generalized association rules”, Proceedings of International Conference on Very Large Data Bases 1995, pages 407-419.

[7] Rakesh Agrawal, Tomasz Imielinski, Arun Swami, “Mining Association Rules between Sets of Items in Large databases”, Proceedings of the 1993 ACM SIGMOD Conference Washington DC, USA, May 1993

[8] Rakesh Agarwal, Ramakrishnan Srikant, “Fast algorithms for mining Association Rules”, In proceedings of the 20th International Conference on Very Large databases, Santigo, 1994, pp 478-499.

[9] S.Y.Wur and Y.Leu, “An effective Boolean Algorithm for mining Association Rules in large databases”, The 6th International conference on Database systems for Advanced Applications, 1999, pp 179-186.

[10] IBM/Quest/Synthetic data.

Comparative Analysis of Serial Decision Tree Classification Algorithms

Matthew N. Anyanwu manyanwu @memphis.edu
Department of Computer Science
The University of Memphis,
Memphis, TN 38152, U.S.A

Sajjan G. Shiva sshiva @memphis.edu
Department of Computer Science
The University of Memphis,
Memphis, TN 38152, U.S.A

Abstract

Classification of data objects based on a predefined knowledge of the objects is a data mining and knowledge management technique used in grouping similar data objects together. It can be defined as supervised learning algorithms as it assigns class labels to data objects based on the relationship between the data items with a pre-defined class label. Classification algorithms have a wide range of applications like churn pre-diction, fraud detection, artificial intelligence, and credit card rating etc. Also there are many classification algorithms available in literature but decision trees is the most commonly used because of its ease of implementation and easier to understand compared to other classification algorithms. Decision Tree classification algorithm can be implemented in a serial or parallel fashion based on the volume of data, memory space available on the computer resource and scalability of the algorithm. In this paper we will review the serial implementations of the decision tree algorithms, identify those that are commonly used. We will also use experimental analysis based on sample data records (Statlog data sets) to evaluate the performance of the commonly used serial decision tree algorithms.

Keywords: Decision tree, Classification Algorithm, data mining, SPRINT, SLIQ, CART, C4.5, IDE3

1. INTRODUCTION

Classification can be described as a supervised learning algorithm in the machine learning process. It assigns class labels to data objects based on prior knowledge of class which the data records belong. It is a data mining techniqueago, has made it possible to co-design and co-develop software and hardware, and hence, such components [1]. However, integration of an that deals with knowledge extraction from database records and prediction of class label from unknown data set of records (Tan et al, 2006). In classification a given set of data records is divided into training and test data sets. The training data set is used in building the classification model, while the test data record is used in validating the model. The model is then used to classify and predict new set of data records that is different from both the training and test data sets (Garofalakis et al, 2000 and Gehrke et al, 1998). Supervised learning algorithm (like classification) is preferred to unsupervised learning algorithm (like clustering) because its prior knowledge of the class labels of data records makes feature/attribute selection easy and this leads to good prediction/classification accuracy. Some of the common classification algorithms used in data mining and decision support systems are: neural networks (Lippmann, 1987), logistic regression (Khoshgoftaar et al, 1999), Decision trees (Quinlan, 1993) etc. Among these classification algorithms decision tree algorithms is the most commonly used because of it is easy to understand and cheap to implement. It provides a modeling technique that is easy for human to comprehend and simplifies the classification process (Utgoff and Brodley, 1990). Most Decision tree algorithms can be implemented in both serial and parallel form while others can only be implemented in either serial or parallel form. Parallel implementation of decision tree algorithms is desirable in-order to ensure fast generation of results especially with the classification/prediction of large data sets, it also exploits the underlying computer architecture (Shafer et al, 1996). But serial implementation of decision algorithm is easy to implement and desirable when small-medium data sets are involved. In this paper we will review the most common decision tree algorithms implemented serially and perform an experiment to compare their classification and prediction accuracy. In Section 2 we did a review of decision tree algorithms, the phases of decision tree construction and its implementation patterns. In Section 3 we review the serial implementation decision tree algorithms and compare their features. In Section 4 we conduct a survey of the implementation statistics of the commonly used serially implemented decision tree algorithms and compare their frequency of usage. In section 5 we perform an experimental analysis of the commonly used decision tree algorithms and evaluate their performance based on execution time and classification accuracy.

2. Decision Tree Algorithm

Decision tree algorithm is a data mining induction techniques that recursively partitions a data set of records using depth-first greedy approach (Hunts et al, 1966) or breadth-first approach (Shafer et al, 1996) until all the data items belong to a particular class. A decision tree structure is made of root, internal and leaf nodes. The tree structure is used in classifying unknown data records. At each internal node of the tree, a decision of best split is made using impurity measures (Quinlan, 1993). The tree leaves is made up of the class labels which the data items have been group. Decision tree classification technique is performed in two phases: tree building and tree pruning. Tree building is done in top-down manner. It is during this phase that the tree is recursively partitioned till all the data items belong to the same class label (Hunts et al, 1966). It is very tasking and computationally intensive as the training data set is traversed repeatedly. Tree pruning is done is a bottom-up fashion. It is used to improve the prediction and classification accuracy of the algorithm by minimizing over-fitting (noise or much detail in the training data set) (Mehta et al, 1996). Over-fitting in decision tree algorithm results in misclassification error. Tree pruning is less tasking compared to the tree growth phase as the training data set is scanned only once. In this study we will review Decision tree algorithms implemented in a serial pattern, identify the algorithms commonly used and compare their classification accuracy and execution time by experimental analysis

3. Serial Implementation of Decision Tree Algorithm

Decision tree algorithm can be implemented in a parallel form based on its scalability with respect to the input data. Parallel implementation tends to be scalable, fast and disk resident and can be implemented in computer architecture with many processors (Shafer et al, 1996). Serial

implementation on the other hand is fast, memory resident and easy to understand. In this paper we will focus on serial implementation of decision tree algorithm by Hunt's algorithms and other serial decision tree algorithms that does not obey Hunt' algorithm (SLIQ and SPRINT). Hunt's method of decision tree construction (Quinlan, 1993 and Hunts et al, 1966) is as stated below: Given a training set T of data records denoted by the classes C= C1, C2, •••, Ck
The decision tree is constructed recursively using depth-first divide-and-conquer greedy strategy by the following cases:
• Case1: T contains all the cases that belong to the same class Cj. The leaf node for T is created and it is known by the class Cj
• Case2: T contains cases that belong to one class or more. The best splitting single attribute is chosen, which will test and split T in to a single-class that contains many cases. The split of T gives the subsets of T which are: T1, T2, •••, Tn. The split on T is chosen in order to obtain mutually exclusive results:

O1, O2, •••, On   Ti T having the result Oi

• Case3: T contains no cases. The leaf created for the decision T has a class from other source which is not T. In C4.5 this is regarded as the most frequent class and is chosen as the parent not of the constructed tree. With Hunt's method decision tree is constructed in two phases: tree growth and pruning phases which have been explained in Section II. Most serial decision tree algorithms (IDE3, CART and C4.5) are based Hunt's method for tree construction (Srivastava et al, 1998). In Hunt's algorithm for decision tree construction, training data set is recursively partitioned using depth-first greedy technique, till all the record data sets belong to the class label (Hunts et al, 1966). The data sets are memory resident and the data sets are sorted at every node in-order to determine the best splitting attribute (Shafer et al, 1996). One of the disadvantages of serial decision tree implementation is low classification accuracy when the training data is large. In order to reduce the high computational complexity associated with large training data set, the whole training data set is loaded into the memory at the same time which leads to low classification accuracy (Srivastava et al, 1998). This short coming of serial decision tree implementation is addressed by SLIQ and SPRINT algorithm. In serial implementation of SPRINT and SLIQ, the training data set is recursively partitioned using breadth-first technique till all the data set belongs to the same class label and there is one time sort of the data set using list data structure. Also the training data set is not memory resident but disk resident, which makes data scalability possible. This approach improves the classification accuracy and reduces misclassification errors. The following sections give a review of the commonly used decision tree algorithms based on serial implementation. Decision trees based on Hunt's algorithm can be classified as classical decision trees which can only be implemented serially. But there have been on-going researches to implement them in a parallel pattern. Peng et al,(n.d) implemented the parallel version of IDE3. The disadvantages associated with classical decision tree algorithms are as enumerated below by Podgorelec et al (2002):
• Handling Noise Data: Classical decision tree algorithms do not always produce decision models with accurate classification when the training data contains noise or too much detail. But C4.5 and enhanced processing technique that handles this deficiency

• Production of Same Type of Decision tree: Given the same training data set and the same condition, classical algorithm always produce the same tree, instead of producing multiple trees with a flexibility to choose the one that is less prone to error.

• Importance of Error: Different errors arises during application of classical decision tree algorithms, but some errors have higher priority than others and need to be minimized to achieve accurate classification. The errors occur as a result of the decisions made in building the tree which reduces classification accuracy.

3.1 IDE3

IDE3 (Iterative Dichotomiser 3) decision tree algorithm was introduced in 1986 by Quinlan Ross (Quinlan, 1986 and 1987). It is based on Hunt's algorithm and it is serially implemented. Like other decision tree algorithms the tree is constructed in two phases; tree growth and tree pruning. Data is sorted at every node during the tree building phase in-order to select the best splitting single attribute (Shafer et al, 1996). IDE3 uses information gain measure in choosing the splitting attribute. It only accepts categorical attributes in building a tree model (Quinlan, 1986 and 1987). IDE3 does not give accurate result when there is too-much noise or details in the training data set, thus a an intensive pre-processing of data is carried out before building a decision tree model with IDE3

3.2 C4.5

C4.5 algorithm is an improvement of IDE3 algorithm, developed by Quinlan Ross (1993). It is based on Hunt's algorithm and also like IDE3, it is serially implemented. Pruning takes place in C4.5 by replacing the internal node with a leaf node thereby reducing the error rate (Podgorelec et al, 2002). Unlike IDE3, C4.5 accepts both continuous and categorical attributes in building the decision tree. It has an enhanced method of tree pruning that reduces misclassification errors due noise or too-much details in the training data set. Like IDE3 the data is sorted at every node of the tree in order to determine the best splitting attribute. It uses gain ratio impurity method to evaluate the splitting attribute (Quinlan, 1993).

3.3 CART

CART (Classification and regression trees) was introduced by Breiman, (1984). It builds both classifications and regressions trees. The classification tree construction by CART is based on binary splitting of the attributes. It is also based on Hunt's model of decision tree construction and can be implemented serially (Breiman, 1984). It uses gini index splitting measure in selecting the splitting attribute. Pruning is done in CART by using a portion of the training data set (Podgorelec et al, 2002). CART uses both numeric and categorical attributes for building the decision tree and has in-built features that deal with missing attributes (Lewis, 200). CART is unique from other Hunt's based algorithm as it is also use for regression analysis with the help of the regression trees. The regression analysis feature is used in forecasting a dependent variable (result) given a set of predictor variables over a given period of time (Breiman, 1984). It uses many single-variable splitting criteria like gini index, symgini etc and one multi-variable (linear combinations) in determining the best split point and data is sorted at every node to determine the best splitting point. The linear combination splitting criteria is used during regression analysis. SALFORD SYSTEMS implemented a version of CART called CART® using the original code of Breiman, (1984). CART® has enhanced features and capabilities that address the short-comings of CART giving rise to a modern decision tree classifier with high classification and prediction accuracy.

3.4 SLIQ

SLIQ (Supervised Learning In Ques) was introduced by Mehta et al, (1996). It is a fast, scalable decision tree algorithm that can be implemented in serial and parallel pattern. It is not based on Hunt's algorithm for decision tree classification. It partitions a training data set recursively using breadth-first greedy strategy that is integrated with pre-sorting technique during the tree building phase (Mehta et al, 1996). With the pre-sorting technique sorting at decision tree nodes is eliminated and replaced with one-time sort, with the use of list data structure for each attribute to determine the best split point (Mehta et al, 1996 and Shafer et al, 1996). In building a decision tree model SLIQ handles both numeric and categorical attributes. One of the disadvantages of SLIQ is that it uses a class list data structure that is memory resident thereby imposing memory restrictions on the data (Shafer et al, 1996). It uses Minimum Description length Principle (MDL) in pruning the tree after constructing it MDL is an inexpensive technique in tree pruning that uses the least amount of coding in producing tree that are small in size using bottom-up technique (Anyanwu et al, 2009 and Mehta et al, 1996).

3.5 SPRINT

SPRINT (Scalable Parallelizable Induction of decision Tree algorithm) was introduced by Shafer et al, 1996. It is a fast, scalable decision tree classifier. It is not based on Hunt's algorithm in constructing the decision tree, rather it partitions the training data set recursively using breadth-first greedy technique until each partition belong to the same leaf node or class (Anyanwu et al, 2009 and Shafer et al, 1996). It is an enhancement of SLIQ as it can be implemented in both serial and parallel pattern for good data placement andload balancing (Shafer et al, 1996). In this paper we will focus on the serial implementation of SPRINT. Like SLIQ it uses one time sort of the data items and it has no restriction on the input data size. Unlike SLIQ it uses two data structures: attribute list and histogram which is not memory resident making SPRINT suitable for large data set, thus it removes all the data memory restrictions on data (Shafer et al, 1996). It handles both continuous and categorical attributes.

4. Serial Decision Tree Algorithm Implementation Statistics

We reviewed about thirty-two articles in order to determine which of the serial decision tree methods is commonly used in practical applications. The outcomes of our literature survey are as stated in the following tables below: the articles are represented by the last name of the first author and the year of publication and also the decision tree algorithm used

Paper	Decision Tree Algorithm
Quinlan, 1983, 1993	IDE3 and C4.5
Shafer et al, 1996	SPRINT, SLIQ, IDE3 and CART
Hunts et al, 1966	CLS, C4.5, CART and IDE3
Breiman, 1984	CART
Fan et al, 2003	Random Tree
Mehta, et al, 1996	SLIQ
Gehrke et al, 1998	RainForest
Peng et al	IDE3
Srivastava et al. 1997	SPRINT, IDE3 and C4.5
Srivastava et al. 1998	SPRINT, IDE3, C4.5 and SLIQ
Rastog et al, 1998	PUBLIC, CLS, IDE3, C4.5 and CART
Sattler and Dunemann, 2001	ID3, C4.5, SPRINT, SLIQ and PUBLIC
Kufrin, 1997	ID3 and CART
BA˘DULESCU, (n.d)	Rainforest, IDE3, C4.5 and SLIQ CART and SPRINT
Srivastava and Singh (n.d)	ID3, C4.5, ClOUDS and SPRINT
Sattler and Dunemann, 2001	ID3, C4.5, SPRINT, SLIQ and PUBLIC
Podgorelec et al, 2002	ID3, C4.5, CART and OCI
Ling et al, 2004	C4.5
Du and Zhan, 2002	ID3 and CART
Pješivac-Grbović et al, 2006	C4.5

Wen et al, 2008	CART and C5.0
Xu et al, 2006	IDE3 and IDE3+

Table 1: Literature review of Decision Tree Algorithms

Table 1 show a literature of decision tree algorithms that is implemented serially.
Table 2 shows the frequency usage of the serial implementation of decision tree algorithms. Table 2 shows that IDE3 is the most frequently used classifier, followed by C4.5 and then SPRINT. ID3 was one the earliest classifiers but as researchers and scientists discovered its flaws they switch to CART, C4.5 and SPRINT

5. Experimental Analysis

We carried out some experiments using Statlog data sets (Michie et al, 1994) as shown in Table 3. The Stalog data set include large scale data sets of various disciplines like financial (Australian and German data sets); transportation (Vehicle data sets), science (Shuttle data set) and health (Heart data set). We did a performance evaluation of the decision tree classifiers. Number of records, number of attributes and class size of the different data sets are varied in-order to determine their effect on the performance of each classifier. Tables 4, 5, and figures (1), (2) shows the result of the analysis.

Decision Tree Algorithm	Frequency Usage
CLS	9%
IDE	68 %
IDE3+	4.5 %
C4.5	54.55 %
C5.0	9%
CART	40.9 %
Random Tree	4.5 %
Random Forest	9%
SLIQ	27.27 %
PUBLIC	13.6 %
OCI	4.5 %
CLOUDS	4.5 %
SPRINT	31.84 %

Table 2: Frequency use of Decision Tree Algorithm

5.1 Experimental Results

Figure (1) shows that for all the classifiers the execution time increases as the number of records increases. The steady portion of the graph is the effect of varying the number of attributes of the classifiers. Also Figure (2) shows that the execution time (time to build the model) of the classifiers decreases as the attributes of the classifiers increases and becomes steady at some point due to the change in the number of records of the data sets and also class size change. Table 4 shows that SPRINT classifiers have the fastest execution time among all the classifiers, irrespective of the class size, number of attributes and records of the data sets volume. This is closely followed by C4.5. The table also showed that execution time for IDE3 is faster that CART but CART is preferred by researches and scientists as it handles both categorical and continuous attributes while IDE3 does not handle continuous attributes. Table 5 shows that SPRINT classifier has the highest classification accuracy among all the classifiers, this is followed by C4.5. The class size, attribute number and record number do not affect the classification accuracy of SPRINT and C4.5 compared to other classifiers. The classification accuracy of the IDE3 and CART classifiers depends to a large extent the class size, attribute number and record number of the data sets. As shown in Table 5 for a large data set (shuttle data set), the classification accuracy of IDE3 is better than that of CART as ID3 has a high accuracy for large data that have been pre-processed (noise and outliers removed) and loaded into the memory at the same time. But for other data sets (Vehicle, Australian, German and Heart) that are not too large (small-medium data sets), the classification accuracy of CART is more than that of IDE3.

Dataset	Category	No. of Attributes	No. of Classes	No. of Records
Australian	Credit Analysis	14	2	690
Shuttle	Space Shuttle Radiation	9	7	43499
German	Credit Analysis	24	2	1000
Heart	Heart Disease Screening	13	2	270
Vehicle	Vehicle Identification	18	4	753

Table 3: Statlog Datasets

Dataset	IDE3	CART	C4.5	SPRINT
Austrilian	0.08secs	11.41secs	0.02secs	0.02secs
Shuttle	1.48secs	38.31secs	0.17secs	0.15secs
German	0.03secs	2.17secs	0.06secs	0.04secs
Heart	0.03secs	0.61secs	0.1secs	0.03secs
Vehicle	0.03secs	1.64secs	0.1secs	0.02secs

Table 4: Execution Time to build Model

Dataset	IDE3	CART	C4.5	SPRINT
Austrilian	71.5 %	85.4 %	84.2 %	85.8 %
Shuttle	99.2 %	94 %	98.00 %	99.63 %
German	32.1 %	70 %	69.2 %	70 %
Heart	35.2%	56.67%	76.7%	80%

Vehicle 54.3% 65% 66.5% 67%

Table5: Classification Accuracy

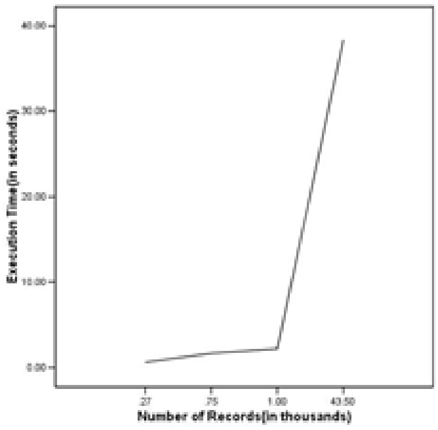


Figure 1: Execution time and Number of Records

6. Conclusion and Future Work

Decision tree induction is one of the classification techniques used in decision support systems and machine learning process. With decision tree technique the training data set is recursively partitioned using depth- first (Hunt's method) or breadth-first greedy technique (Shafer et al , 1996) until each partition is pure or belong to the same class/leaf node (Hunts et al, 1966 and Shafer et al , 1996). Decision tree model is preferred among other classification algorithms because it is an eager learning algorithm and easy to implement. Decision tree algorithms can be implemented serially or in parallel. Despite the implementation method adopted, most decision tree algorithms in literature are constructed in two phases: tree growth and tree pruning phase. Tree pruning is an important part of decision tree construction as it is used improving the classification/prediction accuracy by ensuring that the constructed tree model does not overfit the data set (Mehta et al, 1996). In this study we focused on serial implementation of decision tree algorithms which are memory resident, fast and easy to implement compared to parallel implementation of decision that is complex to implement. The disadvantages of serial decision tree implementation is that it is not scalable (disk resident) and its inability to exploit the underlying parallel architecture of computer system processors. Our experimental analysis of performance evaluation of the commonly used decision tree algorithms using Statlog data sets (Michie et al, 1994) shows that there is a direct relationship between execution time in building the tree model and the volume of data records. Also there is an indirect relationship between execution time in building the model and attribute size of the data sets. The experimental analysis also shows that SPRINT and C4.5 algorithms have a good classification accuracy compared to other algorithms used in the study. The variation of data sets class size, number of attributes and volume of data records is used to determine which algorithm has better classification accuracy between IDE3 and CART algorithms. In future we will perform experimental analysis of commonly used parallel implementation tree algorithms and them compare it that serial implementation of decision tree algorithms and determine which one is better, based on practical implementation.

References

[1] Anyanwu, M., and Shiva, S. (2009). Application of Enhanced Decision Tree Algorithm to Churn Analysis. 2009 International Conference on Artificial Intelligence and Pattern Recognition (AIPR-09), Orlando Florida

[2] BA˘DULESCU, L. A. (n.d). Data mining algorithms based on decision trees. Annals of the ORADEA
university. From
http://www.imtuoradea.ro/auo.fmte/files2006/MIE
files/Laviniu%20Aurelian%20Badulescu%201.pdf Retrieved date: May 13, 2009

[3] Breiman, L., Friedman, J., Olshen, L and Stone, J. (1984). Classification and Regression trees. Wadsworth
Statistics/Probability series. CRC press Boca Raton, Florida, USA.

[4] Du, W., Zhan, Z. (2002). Building decision tree classifier on private data, Proceedings of the IEEE international
conference on Privacy, security and data mining, pp.1-8, Maebashi City, Japan

[5] Garofalakis, M., Hyun, D., Rastogi, R. and Shim, K. (200). Efficient algorithms for constructing decision
trees with constraints. Proceedings of the sixth ACM SIGKDD international conference on Knowledge
discovery and data mining. pp. 335 - 339

[6] Gehrke, J., Ramakrishnan, R., Ganti, V. (1998). RainForest - a Framework for Fast Decision Tree Construction
of Large Datasets.Proceedings of the 24th VLDB conference, New York, USA. pp.416-427

[7] Gehrke, J., Ramakrishnan, R., Ganti, V. (1998). RainForest - a Framework for Fast Decision Tree Construction
of Large Datasets, Proceedings of the 24th VLDB conference, New York, USA. pp.416-427

[8] Hunt, E.B., Marin. and Stone,P.J. (1966). Experiments in induction, Academic Press, New York.

[9] Khoshgoftaar, T.M and Allen, E.B. (1999). Logistic regression modeling of software quality. International Journal of Reliability, Quality and Safety Engineering, vol. 6(4, pp. 303-317.

[10] Kufrin, R. (1997). Decision trees on parallel processors. In J. Geller, H. Kitano, and C. B. Suttner, editors,
parallel Processing for Artificial Intelligence 3 Elsevier Science

[11] Lewis, R.J. (200). An Introduction to Classification and Regression Tree (CART) Analysis. 2000 Annual
Meeting of the Society for Academic Emergency Medicine, Francisco, California

[12] Ling, X. C., Yang, Q., Wang, J and Zhang, S. (2004). Decision trees with minimal costs. Proceedings of the
21st International Conference on Machine Learning, Banff, Canada

[13] Lippmann, R. (1987). An Introduction to computing with neural nets. IEEE ASSP Magazine, vol. (22)

[14] Mehta, M., Agrawal, R., and Rissanen, J. (1995). MDL-based decision tree pruning. International conference
on knowledge discovery in databases and data mining (KDD-95) Montreal, Canada

[15] Mehta, M., Agrawal, R., and Rissanen, J. (1996). SLIQ: A fast scalable classifier for data mining. In EDBT
96, Avignon, France

[15] Michie, D., Spiegelhalter, D., J., and Taylor, C., C. (1994). Machine Learning, Neural and Statistical Class-

fication, Ellis Horword
[16] Peng, W., Chen, J. and Zhou,H. An Implementation of IDE3 Decision Tree Learning Algorithm. From
web.arch.usyd.edu.au/ wpeng/DecisionTree2.pdf Retrieved date: May 13, 2009
[17] Podgorelec, V., Kokol, P., Stiglic, B., Rozman, I. (2002). Decision trees: an overview and their use in
medicine, Journal of Medical Systems Kluwer Academic/Plenum Press, vol.26, Num. 5, pp.445-463.
[18] Pješivac-Grbović, J., Angskun, T., Bosilca, G., Fagg, G.E., Dongarra, J. J. (2006). Decision trees and MPI
collective algorithm selection problem. Tech. Rep. UT-CS-06-586, The University of Tennessee at Knoxville,
Computer Science Department. From < http : //www.cs.utk.edu/library/2006.html >
[19] Peng, W., Chen, J., and Zhou., H (n.d). An Implementation of ID3–– decision tree learning algorithm.
University of New South Wales, School of Computer Science and Engineering, Sydney, NSW 2032, Australia

[20] Quinlan, J. R. (1986). Induction of decision trees. Machine Leaning, vol (1), pp.81-106
[21] Quinlan, J. R. (1987). Simplifying decision trees, International Journal of Machine Studies, number27,
pp. 221-234.

[22] Quinlan, J. R. (1993). C45: Programs for Machine Learning. Morgan Kaufmann, San Mateo, CA.
[23] Utgoff, P and Brodley, C. (1990). An Incremental Method for Finding Multivariate Splits for Decision Trees,
Machine Learning: Proceedings of the Seventh International Conference pp.58.
[24] Rastogi, R., and Shim, K. (1998). PUBLIC: A decision tree classifier that integrates building and pruning.
Proceedings of the 24th VLDB Conference. New York. pp. 404-415
[25] Sattler, K and Dunemann, O. (2001).SQL database primitives for decision tree classifiers Proceedings of the
tenth international conference on Information and knowledge management, pp.379-386
[26] Shafer, J., Agrawal, R., and Mehta, M. (1996). Sprint: A scalable parallel classifier for data mining.
Proceedings of the 22nd international conference on very large data base. Mumbai (Bombay), India
[27] Srivastava, A., Singh, V., Han, E., and Kumar, V .(1997). An efficient, scalable, parallel classifier for data
mining . University of Minnesota, Computer Science, technical report, USA.
[28] Srivastava, A., Singh, V. (n.d). Methods to reduce I/O for decision tree classifiers. IBM T.J Watson Research
center.

[29] Srivastava, A., Singh,V., Han,E., and Kumar, V .(1998). Parallel Formulations of Decision-Tree Classification
Algorithms. Data Mining and Knowledge Discovery, an international journal, pp.237-261

[30] Tan, P., Steinbach, M. and Kumar, V. (2006). Introduction to Data
[31] Wen, X., Hu, G., Yang, X. (2008). CBERS-02 Remote Sensing Data Mining Using Decision Tree Algorithm,
First International Workshop on Knowledge Discovery and Data Mining, pp.86-89

[32] Xu, M., Wang, J. and Chen, T. (2006). Improved decision tree algorithm: ID3+, Intelligent Computing in
Signal Processing and Pattern Recognition, Vol.345, pp.141-149

Secure Tropos: An agent oriented software engineering methodology for the development of health and social care information systems.

Haralambos Mouratidis haris@uel.ac.uk
CITE, UEL, University Way,
Docklands, London

Abstract

A huge amount of health and social care related information needs to be stored and analysed; with the aid of computer systems this can be done faster and more efficiently. As a result, computerised health and social care information systems are gaining popularity. The development of such systems, mostly so far, follows an ad-hoc pattern. However, in order to effectively deal with the characteristics of such systems such as size, security, unpredictability, and openness; appropriate software engineering methodologies and paradigms should be employed for their development. This paper defines a set of requirements that a methodology should demonstrate and it argues for the appropriateness of the Secure Tropos agent oriented methodology for the development of health and social care information systems. The applicability of the methodology is demonstrated with the aid of a real-life case study, the electronic Single Assessment Process system, an information system to support integrated assessment of the health and social care needs of older people. The application of the proposed methodology on the case study indicated that the methodology satisfies the identified requirements.

Keywords: secure health and social care information systems development, methodologies, software agents, Secure Tropos

6.1 INTRODUCTION

Almost every nation is facing problems (managerial, administrative, and/or clinical) with the delivery of health and social care to its citizens [1]. It has been argued [2] that information technology can provide the answer to some of these problems.

Consider as an example the problems associated with the health and social care of older people such as duplication of assessments, lack of awareness of key concepts of need and fragmentation of care. To overcome these problems national policy in England is the development of the Single Assessment Process (SAP). The Single Assessment Process [3] aims to create closer working for providing primary health and social care for older people and other groups. In a distributed health care setting different health care professionals such as general practitioners and nurses, must cooperate to provide older persons with appropriate care and they must also work closely with social care professionals, such as social workers, because health and social care needs overlap amongst older people. Moreover, a huge amount of data needs to be available to the appropriate professionals to allow them to take the correct decisions.

It has been recognised [3] that computerising this process is a very important step for the success of the single assessment process. Computer systems will help to automate some of the tasks involved, such as the management of the health and social care teams, and the appointments' procedures; as well as allowing to store and analyse important information faster and more efficiently, thus giving health and social care professionals more time for the actual care of the older people.

Because of the advantages introduced by the use of computerised systems, such as the ones described above, there is a tendency to develop computerised information systems for every area of the health and social care sector, from systems to link General Practitioners' information to systems which provide support to care assistants.

However, such systems demonstrate an inevitable complexity due to their characteristics, such as size (due to the large amount of data and information that needs to be stored as well as the large amount of processes involved), security (privacy of health and social care information has been widely recognised as one of the important issues for the health and social care sector), unpredictability (the health and social care sector is by nature unpredictable due to its dynamic environment), and openness (due to their wide interconnection with other systems, such as police registers). Therefore, system developers must employ software engineering methodologies to assist them during the development of a system by providing, for instance, methods to elicit the system requirements and develop a design that meets these requirements.

Nevertheless, most of the times, an ad-hoc approach is taken for the development of health and social care information systems, partially because current software engineering methodologies are not well suited for the development of such systems. First of all, they fail to deal effectively with the complexity introduced by these systems. Secondly, they fail to provide a clear ontology, which will allow both system developers and users/stakeholders to actively contribute to the development of a system. Thirdly, they fail to consider security as an integral part of the development process.

Therefore, it is important to develop methodologies that will overcome the above limitations, and will allow the development of health and social care information systems in an effective and efficient manner. In doing so, it is important to define the requirements that a methodology for health and social care information systems should demonstrate.

This paper defines such requirements and it presents Secure Tropos, an agent oriented methodology, which is an extension of the Tropos methodology [14]. Tropos demonstrates characteristics that make it ideal for the development of health and social care information systems. Firstly, it allows a deeper understanding of not only the system, but also of the environment in which the system would operate. Secondly, Tropos employs the same concepts and notations throughout the development process leading to a homogeneous development that is easier to understand by non-software engineering experts (such as the health and social care professionals). Thirdly, it employs real life concepts and therefore it narrows the gap between the concepts used by the health and social care professionals and the system developers. Moreover, Secure Tropos adds a fourth important characteristic by considering security issues as an integral part of its development process; therefore leading to the development of secure systems, something very important in the case of health and social care information systems.

The paper is structured as follows. Section 2 defines some important requirements that a software engineering methodology should demonstrate to be suitable for the development of health and social care information systems. Section 3 provides an introduction to the agent oriented software engineering paradigm, necessary for the readers to understand the rest of the paper. Section 4 discusses the appropriateness of the Secure Tropos methodology in the development of health and social care information systems, indicating why Secure Tropos is more suitable, than other methodologies, for the development of these systems. Section 5 presents the methodology, whereas Section 6 demonstrates its applicability with the aid of the electronic Single Assessment Process (eSAP) system case study. Finally, section 7 presents related work and section 8 concludes the paper and describes areas of future work.

The requirements of a software engineering methodology for the development of health and social care information systems.

During the development of, health and social care or otherwise, information systems, software engineers employ guidelines, notations and follow structure processes to help them go through the development faster and more efficiently. In other words, they employ methodologies. To emphasise the need of a methodology Birrell and Ould argue “anyone undertaking software development, on no matter what scale, must be strongly advised to establish a methodology for that development –one or more techniques that, by careful integration and control, will bring order and direction to the production process” [4].
According to Booch [5] “a methodology is a collection of methods applied across the software development life cycle and unified by some general, philosophical approach”. Thus a methodology is considered to be a pre-defined series of steps that helps designers to understand a problem and model a solution. Although a methodology must guide through its steps, on the same time it must be flexible and allow creativity.
Moreover, there are requirements that methodologies specifically employed for the development of health and social care information systems, should demonstrate. As argued by Norris [1], “the term healthcare is a deceptively simple one embracing multiple, inherently complex, sets of processes and interactions”. As a result, health and social care information systems are, usually, large complex systems. Therefore, the development of such systems implies the decomposition of the main system to smaller interrelated subsystems, each of which is also decomposed to smaller subsystems, until the lowest element of the system is defined. Therefore:

Requirement 1:

An appropriate methodology should provide concepts and procedures to decompose effectively the system to smaller, easier to understand and develop components.
Moreover, due to the dynamic and unpredictable nature of the health and social care infrastructures, the relationships between the subsystems might change over time, and new subsystems might often be added into (or removed from) the main system. Therefore:

Requirement 2:

An appropriate methodology should allow developers to iterate during the development stages and easily modify/add/delete the various components of the system.
In addition, the description of health and social care systems usually takes place by identifying the relationships between the stakeholders/users and their interaction with the system. However, there is a fundamental difficulty in eliciting these relationships due to the abstraction gap between the concepts used by health and social care professionals to describe the functionalities of the system, and the software engineers to model such functionalities. Moreover, as a result of the difficulty to understand/been explained the concepts involved, health and social care professionals, usually, are not actively involved in the development of the system under development resulting in the development of a system that does not actually meet the needs of its users. Therefore:

Requirement 3*:*

An appropriate methodology should provide concepts that are easily understood not only by the system developers but also from the system’s stakeholders/users.
Most of the developers have limited knowledge of the health and social care sector, and as a result, and due to the limited involvement of health and social care professionals, find it difficult to understand the implications of their analysis and designs. Therefore:

Requirement 4:

An appropriate methodology should allow developers to model not only the system but also the environment in which the system will be deployed as well as the relationships between the different stakeholders/users of the system.

Additionally, security is an important issue when developing health and social care information systems as such systems contain sensitive information which should be kept private. It has been argued in the literature [6,7] that in order to identify potential conflicts, which could lead to vulnerabilities, between security and the functionalities of the system; security should be considered from the early stages of the system development. Therefore:

Requirement 5:

An appropriate methodology should allow developers to consider security issues throughout the development stages by providing concepts and processes customised to security.

Thus, to demonstrate the suitability of the Secure Tropos methodology in the development of health and social care information systems, we must demonstrate the degree at which Secure Tropos meets the above requirements. However, before doing this, it is important to provide a brief overview of agent oriented software engineering, to enable readers with limited knowledge of it, to understand the paradigm in which Secure Tropos is based. The next section provides such an overview.

6.2 Agent Oriented Software Engineering

As mentioned above, Tropos is based on the agent oriented paradigm. Agent oriented software engineering is based on the concept of an agent1. The term agent derives from the present particle of the Latin verb agere, which means to drive, act, lead or do [8]. One of the most widely used definition of an agent, by Wooldridge and Jennings [9], defines it as "…. A hardware or (more usually) software based computer system that enjoys the following properties:

1. **Autonomy**. Agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state.
2. **Social ability.** Agents interact with other agents (and possibly humans) via some kind of agent communication language.
3. **Reactivity**. Agents perceive their environment, (which may be the physical world, such as a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it.

Pro-activeness.

Agents do not simply act in response to their environment; they are able to exhibit goal-directed behaviour by taking the initiative".Agent Oriented Software Engineering introduces an alternative approach, than other software engineering approaches, in analysing and designing complex distributed computerised systems [10,11,12]. According to this, a complex computerised system is viewed as a multiagent system [11] in which a collection of autonomous software agents (subsystems) interact with each other in order to satisfy their design objectives. Therefore, developers view the system as a society, similar to a human society, consisting of entities that possess characteristics similar to humans such as mobility, intelligence and the capability of communicating.

6.3 The arguments for the suitability of Secure Tropos for the development of health and social care information systems.

It is important, when arguing for the suitability of the Secure Tropos methodology for the development of health and social care information systems, to discuss agent orientation with respect to the requirements defined in Section 2.

1 In this paper the term agent refers always to a software agent.

According to Jennings and Wooldridge [13] an agent oriented approach is effective in partitioning the problem space of a complex system. A complex computerised system can be decomposed into smaller components, the same way that a multiagent system can be decomposed into the elements that constitute the system (software agents). As mentioned above, when adopting an agent oriented view of a system, the system is viewed as a society similar to a human society, consisting of entities that possess characteristics similar to humans such as mobility, intelligence and the capability of communicating. This allows decomposing the system easier by mapping the real life counterparts of a system to a software subsystem (agent).
Moreover, agent orientation is well suited for the development of dynamic environments, in which the structure of the system changes. Agent structures provide a variety of stable intermediate forms, meaning that individual agents or organisational grouping can be developed in relative isolation and then added into (or removed from) the system in an incremental manner [13].
In addition, agent oriented software engineering provides high-level abstractions, such as agents, (social) interactions and organisations, in modelling complex systems. This higher (than other software paradigms) level of abstraction employed in the development of software systems results in modelling a system in terms of autonomous entities with characteristics similar to humans. This introduces a close-to-real-life modelling of the system and therefore makes the development of the software system natural. In turn, this helps to narrow the gap between real life and modelling, by allowing developers to reason about the software system using concepts and mental qualities known to them from the real life. This is particularly true in the case of health and social care information systems, where the high level of abstraction, and the usage of concepts close to real life – such as goals, tasks, actors- leads to a better mutual understanding between the system developers (computer scientists) and system users (health and social care professionals) since system developers can better explain the functionalities of the system, by decomposing it to smaller autonomous entities (agents) that posses characteristics similar to humans, such as mobility and the ability to communicate, and the system users can use the concept of an agent to describe more precisely the needs of the system.
Additionally, the agent oriented software engineering paradigm presents a feasible approach for the integration of security to software engineering. This is mainly due to the appropriateness of agent oriented philosophy for dealing with the security issues that exist in a computer system. Security requirements are mainly obtained by analysing the attitude of the organisation towards security and after studying the security policy of the organisation. As mentioned in [13] agents act on behalf of individuals or companies interacting according to an underlying organisation context. The integration of security within this context will require for the rest of the subsystems (agents) to consider the security requirements, when specifying their objectives and interactions therefore causing the propagation of security requirements to the rest of the subsystem. In addition, the agent oriented view is perhaps the most natural way of characterising security issues in software systems. Characteristics, such as autonomy, intentionality and sociality, provided by the use of agent orientation allow developers first to model the security requirements in high-level, and then incrementally transform these requirements to security mechanisms.
On the other hand, Secure Tropos in particular demonstrates some key features that make it ideal for the development of health and social care information systems. Firstly, it covers the early stages of requirements analysis [14], and thus allows for a deep understanding of not only the system itself, but also of the environment where the system will operate and as a result it helps to better understand the interactions and the relationships that will occur in the system. This is, as discussed above, very important in the development of health and social care information systems since it allows developers to obtain a clear idea of the environment in which the system will be deployed as well as the interactions between the different health and social care professionals involved and as a result develop a system which is customised to that particular environment and meets the necessities of its users.
Secondly, Secure Tropos employs concepts such as stakeholders, actors, goals, tasks, capabilities, and dependencies which are concepts used in real world situations. This results in introducing during the development process concepts familiar to both users/stakeholders and system developers, and therefore minimise the abstraction gap between the concepts used by health and social care professionals and software engineers when describing a system.
Moreover, Secure Tropos uses the same concepts and notations throughout the development stages. This leads to a uniform development that enables system users (health and social care professional in our case) to be involved throughout the development stages and not only during the requirements elicitation stage.

Another important key feature of Secure Tropos is the consideration of security issues during the development of the system. The security requirements of the various health and social care professionals involved with the system are identified with the aid of security related concepts such as security constraints and secure goals. This enables system developers to understand the security requirements of the system by analysing the security related concepts imposed to the various stakeholders/ users of the system. Then, these requirements can be transformed to (security related) functionalities that the system has to satisfy.
To validate our claims, we have applied Secure Tropos to the development of the electronic Single Assessment Process (eSAP) system case study. However, before illustrating this, the next section provides an overview of the concepts and the development stages of the methodology for readers not familiar with it.

6.4 Description of Secure Tropos

Secure Tropos is an extension of the Tropos methodology. Tropos2 is a novel agent oriented software engineering methodology tailored to describe both the organisational environment of a multiagent system and the system itself, emphasizing the need to understand not only what organisational goals are required, but also how and why the intended system would meet the organisational goals. This allows for a more refined analysis of the system dependencies, leading to a better treatment not only of the system's functional requirements but also of its non-functional requirements, such as security, reliability, and performance [15].
Tropos supports the idea of building a model of the system that is incrementally refined and extended from a conceptual level to executable artefacts, by means of a sequence of transformational steps [16]. Such transformations allow developers to perform precise inspections of the development process by detailing the higher level notions introduced in the previous stages of the development. In addition, since the methodology employs the same notation throughout the development process, such a refinement process is performed in a more uniform way as compared, for example, to UML-based methodologies where the graphical notation changes from one development step to another (for example, from use cases to class diagrams). Moreover, Tropos allows developers to consider security issues as an integral part of the system development. The security process is one of analysing the security needs of the stakeholders and the system in terms of security constraints imposed to the system and the stakeholders; identify secure entities that guarantee the satisfaction of the security constraints and assign capabilities to the system to help towards the satisfaction of the secure entities.
In Secure Tropos, a security constraint is defined as a restriction related to security issues, such as privacy, integrity and availability, which can influence the analysis and design of a multiagent system under development by restricting some alternative design solutions, by conflicting with some of the requirements of the system, or by refining some of the system's objectives [17].
A security constraint contributes to a higher level of abstraction, meaning that security constraints do not represent specific security protocol restrictions3, which restrict the design with the use of a particular implementation language. This higher level of abstraction allows for a generalised design free of models biased to particular implementation languages. Regarding the constraint metamodel, a security constraint is captured through a specialisation of constraint into the subclass security constraint.
In addition, Tropos adopts the i* modelling framework [18], which uses the concepts of actors, goals and social dependencies for defining the obligations of actors (dependees) to other actors (dependers), and the novelty of the methodology lays on the fact that those concepts are used to model not just early requirements, but also late requirements as well as architectural and detailed design [19]. Using the same concepts during the development stages of a multiagent system provides the advantage of reducing impedance mismatches between different development stages, and therefore streamlines the development process [19].
The system and its environment are viewed as a set of actors, who depend on other actors to help them fulfil their goals. An actor [18] represents an entity that has intentionality and strategic goals within the

[2] The name Tropos derives from the Greek "Τρόπος" which means "way of doing things" but also has the connotation of "easily changeable, easily adaptable".

[3] Such security restrictions should be specified during the implementation of the system and not during the analysis and design.

multiagent system or within its organisational setting. An actor can be a (social) agent, a position, or a role. Agents can be physical agents, such as a person, or software agents.

A goal [18] represents a condition in the world that an actor would like to achieve. In Tropos, the concept of a hard-goal (simply goal hereafter) is differentiated from the concept of soft-goal. A soft-goal is used to capture non-functional requirements of the system, and unlike a (hard) goal, it does not have clear criteria for deciding whether it is satisfied or not and therefore it is subject to interpretation [18]. For instance, an example of a soft-goal is “the system should be scalable”. On the other hand, a secure goal represents the strategic interests of an actor with respect to security. Secure goals are mainly introduced in order to achieve possible security constraints that are imposed to an actor or exist in the system. However, a secure goal does not particularly define how the security constraints can be achieved, since alternatives can be considered.

A task represents, at an abstract level, a way of doing something [15]. The fulfilment of a task can be a means for satisfying a goal, or for contributing towards the satisficing of a soft-goal. Secure Tropos allows reasoning about the different ways the actors can achieve their goals, by enabling developers to model alternative tasks that actors might employ to achieve their goals. Similarly, a secure task is used to define precisely how a secure goal can be achieved, by representing a particular way of satisfying a secure goal.

A resource [15] presents a physical or informational entity that one of the actors requires. The main concern when dealing with resources is whether the resource is available and who is responsible for its delivery. On the other hand, a secure resource can be defined as an informational entity that is related to the security of the information system. Secure resources can be divided into two main categories. The first category contains secure resources that display some security characteristics, imposed by other entities, such as security constraints, secure goals, secure tasks and secure dependencies. The second category of secure resources involves resources directly associated with the security of the system. For example, consider the authorisation details file of a component of the system.

A dependency [18] between two actors represents that one actor depends on the other to attain some goal, execute a task, or deliver a resource. The depending actor is called the depender and the actor who is depended upon is called the dependee. The type of the dependency – goal, task, resource- describes the nature of an agreement (called dependum) between dependee and depender. By depending on the dependee for the dependum, the depender is able to achieve goals that it is otherwise unable to achieve on their own, or not as easily or not as well [18]. Moreover, a secure dependency [17] introduces security constraint(s) that must be fulfilled for the dependency to be satisfied. Both the depender and the dependee must agree for the fulfilment of the security constraint in order for the secure dependency to be valid. That means the depender expects from the dependee to satisfy the security constraint(s) and also that the dependee will make an effort to deliver the dependum by satisfying the security constraint(s). There are three different types of a secure dependency: Dependee Secure Dependency, in which the depender depends on the dependee and the dependee introduces security constraint(s) for the dependency. The depender must satisfy the security constraints introduced by the dependee in order to help in the achievement of the secure dependency. Depender Secure Dependency, in which the depender depends on the dependee and the depender introduces security constraint(s) for the dependency. The dependee must satisfy the security constraints introduced by the depender otherwise the security of the dependency will be in risk. Double Secure Dependency, in which the depender depends on the dependee and both the depender and the dependee introduce security constraints for the dependency. Both must satisfy the security constraints introduced to achieve the secure dependency.

A capability [15] represents the ability of an actor of defining, choosing and executing a task for the fulfilment of a goal, given certain world conditions and in presence of a specific event. A secure capability represents the ability of an actor/agent to achieve a secure goal, carry out a secure task and/or deliver a secure resource.

A graphical representation of the Tropos concepts is illustrated in Figure 1.

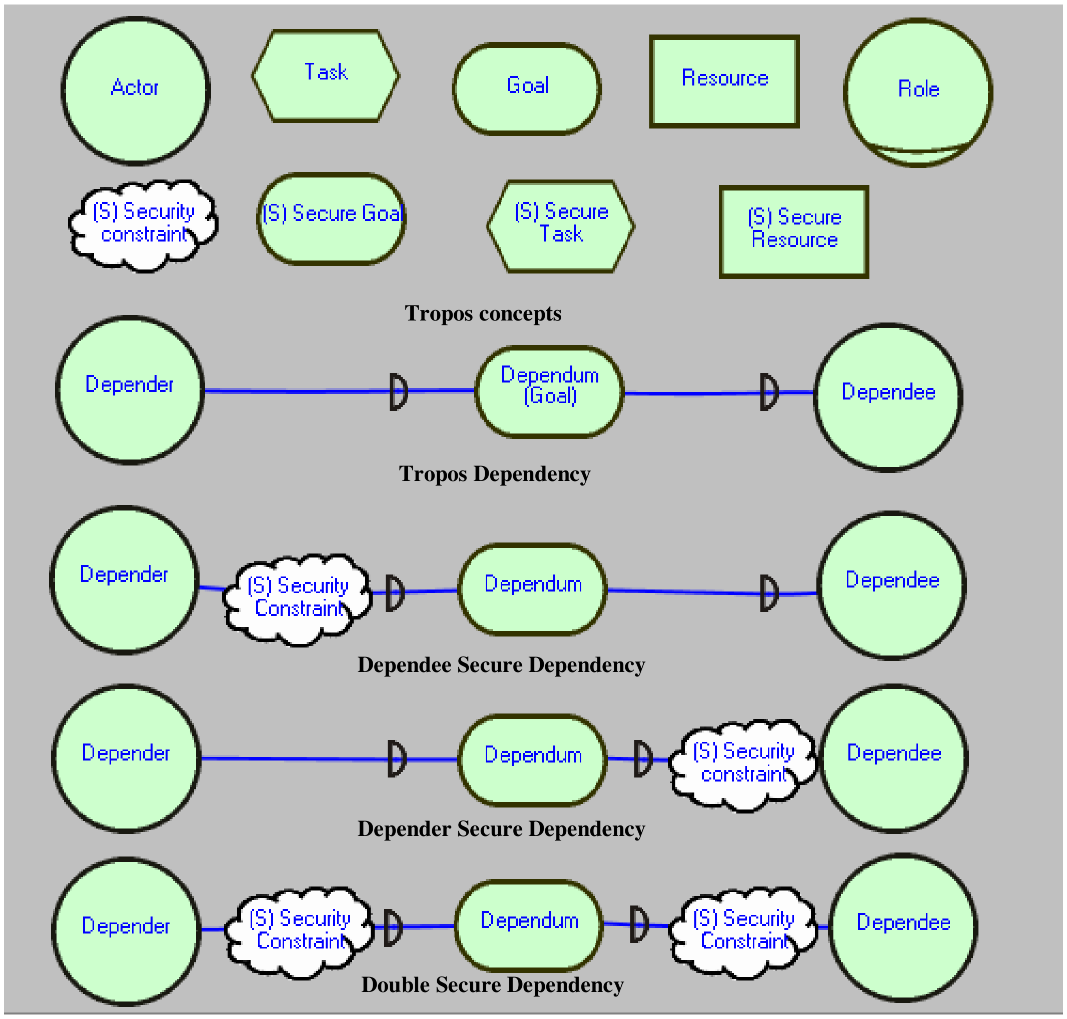


Figure 1: Tropos concepts graphically

Tropos defines its own modelling language [14] in terms of a UML metamodel. The Tropos metamodel is organised into four levels. The meta-metamodel level, which provides the basis for metamodel extensions; the metamodel level, which provides constructs for modelling knowledge level entities and concepts; the domain level, which contains a representation of entities and concepts of a specific application domain; and the instance level, which contains instances of the domain level. For instance, consider an entity as an example of the meta-metamodel, an actor as an example of the metamodel level, a doctor as an example of the domain level and John as an example of the instance level. In addition, the meta-metamodel level of the language allows the inclusion of constructs for the formal definition of the Tropos concepts. In particular a formal specification language, called Formal Tropos, is under development [20]. Formal Tropos [20,21] offers all the concepts of graphical Tropos, such as actors, goals and dependencies, supplemented with a rich temporal specification language, inspired by KAOS [22].

The Secure Tropos methodology covers four main software development stages. During the early requirements analysis stage, developers are concerned with the understanding of a problem by studying an existing organisational setting. This involves the identification of the domain stakeholders and their modelling as social actors. In particular, developers model the stakeholders as actors, their intentions as goals, and their relationships as dependencies. Through a goal-oriented analysis [16], the actors' goals are decomposed into more precise goals and sometimes into tasks that if performed by the actor, allow for goal achievement. From the security point of view, security constraints are imposed to the stakeholders of the system (by other stakeholders). Such constraints are initially expressed in high level statements, and they are later further analysed and secure goals and entities are introduced

to corresponding actors to satisfy them. The output of this phase is an organisational model, which includes relevant actors and their respective dependencies and security entities.
In the late requirements analysis stage, the system-to-be is specified within its operational environment, together with relevant functions and qualities. This description models the system as an actor, who has a number of dependencies with the actors identified during the previous stage. These dependencies indicate the obligations of the system towards its environment, and therefore define the system's functional and non-functional requirements. Moreover, security constraints are delegated to the system-to-be and appropriate entities are assigned to satisfy them.
During the architectural design stage, the system's global architecture is defined in terms of subsystems, interconnected through data and dependencies. In particular, subsystems are represented as actors and data/control interconnections are represented as (system) actor dependencies. This stage is divided into three steps. The first step includes the definition of the overall architectural organisation by introducing new actors to the system and delegating to them some of the goals of the system. The second step includes the identification of the capabilities needed by the actors to fulfil their goals and tasks and the third step involves the identification of a set of agent types and the assignment of capabilities to those agents. From the security point of view, any possible security constraints and secure entities that new actors might introduce are analysed. Additionally, the architectural style of the multiagent system is defined with respect to the system's security requirements and the requirements are transformed into a design with the aid of security patterns. The final output of this stage is a set of software agents corresponding to the actors of the system, each characterised by its specific capabilities.
In the detailed design stage, each architectural component is defined in further detail in terms of inputs, outputs, control, security and other relevant information. This stage is based on the specifications resulting from the analysis of the previous stages and therefore the reasons for a given element at this stage can be traced back to the early requirements analysis. For this stage, Secure Tropos uses elements of the Agent Unified Modeling Language (AUML) [23] to complement the features of i*.
It must be noted that Tropos is not a "laboratory" methodology but it has been motivated by a number of case studies [14,19].

6.5 Case Study

The assessment of health and social care needs is at the heart of good practice in the care of older people. Older people often have multiple impairments and health problems, and complex support systems involving several health and social care practitioners and family carers. Sharing of assessment information is important to avoid unnecessary repetition and to ensure that all relevant information is available to support effective care planning. Recognition of the need to share assessment information has stimulated standardisation of assessment methods. These in turn have been used to help standardise care planning and referrals following assessment.
In March 2001, the (English) Department of Health published its National Service Framework (NSF) for Older People's Services [3]. The NSF sets national standards for the health and social care of older people.
Standard 2 of the National Service Framework, which refers to person-centred care, includes requirements to establish a single assessment process for integrating the assessment of health and social care needs of older people. Local health and social care communities have to introduce standardised shared systems for assessing needs, with convergence towards a fully integrated and electronically based national system. The Department issued further guidance in February 2002, listing requirements for contact, overview, specialist and comprehensive assessment, and a range of assessment instruments, which could be used for these types of assessment.
Information technology has the potential to improve efficiency and effectiveness in the collection and sharing of assessment information. An information system, called hereafter, the electronic single assessment process (eSAP) system, to support integrated assessment of the health and social care needs of the older person, should therefore build on contact and overview assessment in primary care, with maximum involvement of the older person in prioritising the assessment domains and in care planning.

6.6 A typical scenario

Our aim in this paper is not to provide a complete analysis and design of the electronic single assessment process (eSAP) system, but rather to illustrate how the Secure Tropos methodology can be

effectively used in the development of a health and social care information system. As a result, we focus our illustration to a specific scenario related to the functionality of the electronic single assessment process. This allows us to define well the boundaries of the system, and as a result it makes the understanding of the methodology easier for the readers, since only important issues of the methodology are presented without going into unnecessary details.
The following scenario has been used in this paper for the analysis and design of the electronic single assessment process.
"An 81 years old lady, widow, lives in her house. Her daughter lives nearby but she has children of her own and therefore she is unable to provide full care to her mother. However, she sees her mother everyday.
The daughter visits the mother's General Practitioner (GP) to describe her concern about her mother's health. Her mother has become unsteady on her feet and may have had a number of falls. Single assessment process has been introduced, so the GP asks the daughter to complete the EasyCare [24] contact assessment and the information is entered into the GP's computer. The GP sees the daughter concerned about her mother's health and asks his practice nurse to visit the old lady to perform an overview assessment.
The old lady's information is transferred to the nurse's computer along with referrals and instructions, e.g. the daughter of the patient is concerned about her mother's health so please perform an overview assessment. The nurse receives the information and arranges to visit the old lady by generating and sending a letter to the old lady and her daughter giving details about visit and ask availability. The daughter replies (also provides her mother's response) that the date/time is suitable.
The nurse visits the old lady and completes most of the EasyCare assessment except from the health promotion module. From the evaluation of the other modules the nurse concludes the old lady has a number of problems with her house, which increase the risk of falls, she needs help with dressing and also she is not getting the appropriate financial benefits. Then the nurse asks the old lady if the information can be shared, and the old lady accepts. The nurse then produces a care plan summarising all the problems identified and the actions to be taken. She also makes two referrals one to a Social Worker (SW) - to check for a care assistant to help the old lady with dressing and to check about financial benefits- and a second to an Occupational Therapist (OT) -to perform a house assessment for need and adaptation. She then forwards the care plan and a summary of the problems to the General Practitioner and the care plan and contact information to the Social Worker and the Occupational Therapist. In addition, a copy is produced for both the old lady and her daughter and the care plan is signed.
Later, the old lady is visited by the Occupational Therapist who performs the house assessment and decides that the house needs to be adapted to the old lady's needs. The O.T. then makes a referral to the Equipment Services for equipment and also provides the contact information of the old lady. In addition, the O.T. forwards to the GP, nurse and the S.W. a copy of the house problems, needed equipment and informs them that a referral has been made to the Equipment Services.
The Social Worker visits the old lady and identifies that the old lady must apply for financial benefits. A form is produced, filled in, and sent to the Benefits Agency together with old lady's contact and bank information. In addition, the Social Worker agrees to employ a Care Assistant (C.A.) to help the old lady with dressing. A Care Assistant is identified and the Social Worker asks the old lady if she feels comfortable with the identified Care Assistant and the old lady agrees. Contact and overview assessment information is sent to the Care Assistant by the Social Worker. Also, because of the employment of a Care Assistant, the benefits must be adjusted. The social worker informs the benefits agency about it.
While the Care Assistant visits the old lady, she realises that the health promotion module of the overview assessment is not completed. She notifies the nurse and prompts the old lady to fill in the module. When the module is complete, the C.A. sends the information to the General Practitioner, the nurse and the Social Worker.
One of the observations of the Care Assistant was that the old lady didn't have her blood pressure taken for the last 5 years, so she alerts the nurse and the G.P. The General Practitioner receives the alert and makes a referral to the nurse to go and check the blood pressure of the old lady. The nurse visits the old lady to review all the actions of the care plan and also check the old lady's blood pressure. The care plan is updated and for the time being the old lady gets everything she needs."

6.7 developing the eSAP

The above scenario provides the basis for the development of the system. As mentioned in the previous section the first phase of the Tropos methodology is the early requirements analysis.

6.2.1 Early requirements analysis

During the early requirements analysis the software engineer models the dependencies between the stakeholders (actors). For this reason, the Secure Tropos methodology introduces actor diagrams [14]. In such a diagram, actors (graphically represented as circles4) are modelled together with their goals (represented as ovals), soft-goals (represented as bubbles), security constraints (represented as clouds) and their dependencies (represented as links between the actors indicating the dependum). From the scenario presented above, the following actors are derived:

Older Person: The Older Person actor represents patients aged 65 or above, assessed for their health and social care needs. In the presented scenario, the old lady plays this actor. The Older Person must provide information about their health and social care situation, and also receive information such as a summary of their needs and a copy of their care plan. To provide information, the Older Person must undertake assessments. This requires the Older Person to agree with the health and social care professionals on the date/time that the assessments will take place. In addition, the Older Person must understand the procedures clearly, have access to information regarding their care 24 hours every day, and also decide if their information will be shared between the professionals' (and possibly other people) involved in their care. Also, the Older Person must follow the care plan indicated by the health and social care professionals. Therefore, the help of carers (informal-like the daughter- and paid-like the care assistant-) is required.

Nurse: The Nurse performs the overview assessment to the Older Person. To do this, the Nurse must contact the Older Person and arrange a meeting. After performing the assessment the Nurse identifies the care needs of the Older Person, and according to those needs she provides referrals. Also the Nurse must ask for the older person's consent in order to share information with others who may be involved in the care of the Older Person. She generates a care plan and produces a copy of it for the Older Person. In addition, the Nurse informs everyone involved (taking into account the consent of the Older Person) about the care plan and the condition of the Older Person. The Nurse is also responsible for regular review of the care plan.

General Practitioner: The General Practitioner performs the contact assessment, provides referrals to the Nurse to perform an overview (or any different kind she/he thinks is appropriate) assessment, and provides the older person's contact information. In addition, the General Practitioner receives alerts and information regarding the Older Person, such as the care plan, possible referrals, and updates of the care plan.

Social Worker: The Social Worker receives referrals (indicating the problems occurred) and the actions to be performed, and also information about the Older Person such as contact information and a copy of the care plan. According to the referrals, the Social Worker identifies the needs of the Older Person and takes actions. The Social Worker is usually responsible for identifying a suitable care assistant (if necessary) and also dealing with benefits problems that the Older Person might have. After identifying particular problems the Social Worker provides referrals, informs the other professionals involved in the care of the Older Person and updates the care plan. In addition, the Social Worker manages the care assistant.

[4] For a reminder of the graphical representation of the Tropos concepts please refer to Figure 1.

Secondary Care Professional: Secondary care professionals (or specialists) undertake assessment and care following referral by primary care professionals. Some secondary care professionals such as community psychiatric nurses work at the interface between primary and secondary care. During the single assessment process, secondary care professionals, usually, do specialist and comprehensive assessments. In the presented scenario, the Occupational Therapist plays this role. The Occupational Therapist receives referrals from the Nurse along with the contact and overview assessment information of the Older Person. She performs a specialist assessment and identifies specialist needs of the Older Person. According to the identified needs, the Occupational Therapist provides referrals, informs the other professionals involved in the care of the Older Person and also updates the care plan.

Care Assistant: The main aim of the Care Assistant is to help the Older Person with everyday needs. The Care Assistant receives information about the Older Person, such as contact and overview assessment, and updates any of those if necessary by providing to the Nurse possible needs of the Older Person. In addition, she informs the General Practitioner, the Social Worker and the Nurse of any updates regarding the older person's information.

Informal Carer: Informal carers include unpaid family members, friends, and neighbours who help meet older persons' needs for care and support, including meeting emotional (visiting and support), financial (help with managing bills), domestic (help with shopping) and personal (help with dressing) care needs. In the presented scenario the daughter of the old lady plays this role.

Care Manager: A Care Manager, usually a Social Worker or a Nurse, coordinates the delivery of care to the Older Person and plans the work of the care assistants. In the presented scenario, the Social Worker plays the Care Manager.

Benefits Agency: The Benefits Agency actor represents a financial agency that helps older persons financially.
The dependencies, between the above actors are modelled as shown in Figure 2.

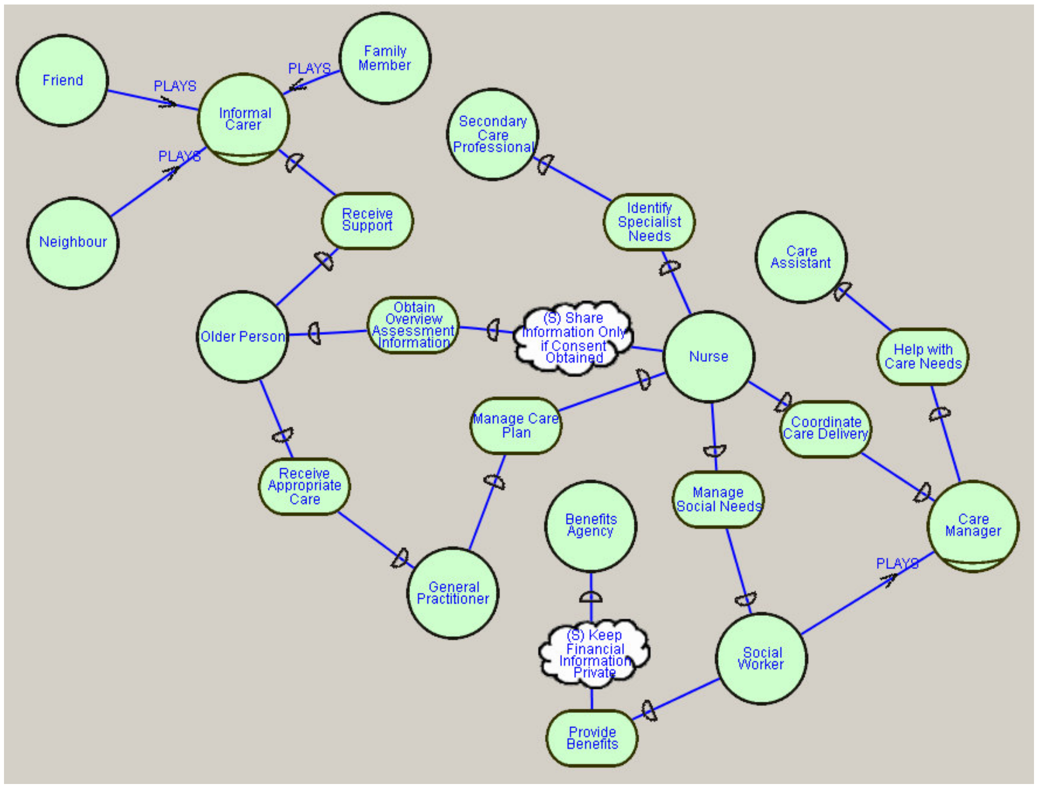


Figure 2: Partial Actor diagram for the eSAP

For instance, the Older Person depends on the General Practitioner to Receive Appropriate Care and on the Informal Carer to Receive Support. On the other hand, the Nurse depends on the Secondary Care Professional to Identify Specialist Needs, on the Care Manager to Coordinate Care Delivery, on the Social Worker to Identify Social Needs and on the Older Person to Obtain Overview Assessment Information.
However, one of the most important and delicate matters for the Older Person is the privacy of their personal medical information and the sharing of it. Therefore, the Older Person imposes a security constraint (share information only if consent Obtained) on the Nurse for the Obtain Overview Assessment Information dependency to be valid. In addition, the Social Worker imposes a security constraint (Keep Financial Information Private) on the Benefits Agency for the Provide Benefits dependency to be valid.
Modelling the dependencies and the security constraints of the individual actors allows developers to model the functional and security requirements of the system according to the real needs of its stakeholders. For example, in the presented analysis, the lack of identifying the security constraints between the Nurse and the Older Person, or the Social Worker and the Benefits Agency would result in a design that would miss important information regarding the security of the system.
When the stakeholders, their goals and the dependencies between them have been identified, the next step of this phase is to analyse in more depth each goal relative to the stakeholder who is responsible for its fulfilment. For this reason, Tropos employs goal diagrams. In a goal diagram, each actor is represented as a dashed-line balloon within which the actor's goals, security constraints and dependencies are analysed. The nodes of the diagram represent goals, soft-goals, tasks, security constraints, and secure entities whereas the links identify the different kinds of relationships between those nodes. Moreover, these links can be connected with external dependencies (identified in the actor diagram) when the reasoning of the analysis goes beyond the actor's boundary [18]. Means-end, contribution and decomposition analysis is used in goal diagrams. Means-end analysis is a form of analysis, consisting of identifying goals, tasks, and/or resources that can provide means for reaching a goal [14]. Contribution analysis is a form of analysis that discovers goals and tasks that can contribute positively or negatively to the achievement of another goal. It is worth mentioning that contributions

could be positive or negative [14]. These are graphically represented with plus (+) and minus (-) signs respectively. When a sign is not used, it is assumed the contribution is positive. Decomposition analysis defines the decomposition of a root goal/ task into sub goals /tasks.
As an example of a goal diagram, consider the Nurse actor shown in Figure 3. The main goal of the Nurse[5] is to Manage the Care Plan. To satisfy this goal the Nurse must Generate the Care Plan, Review the Care Plan and Provide Information. However, the Share Information only if Consent Obtained security constraint imposed to the Nurse by the Older Person (see Figure 2) restricts the Share Older Person Information goal of the Nurse. For the Nurse to satisfy this constraint, a secure goal is introduced Obtain Older Person Consent.
Furthermore, the analysis indicates that the Use of eSAP will enable the Nurse actor to work more efficiently, with less effort, convenient and faster. However, an analysis [17] regarding the security of the eSAP system indicates that Authorisation is required for the eSAP system (in order to help towards the Privacy security feature).

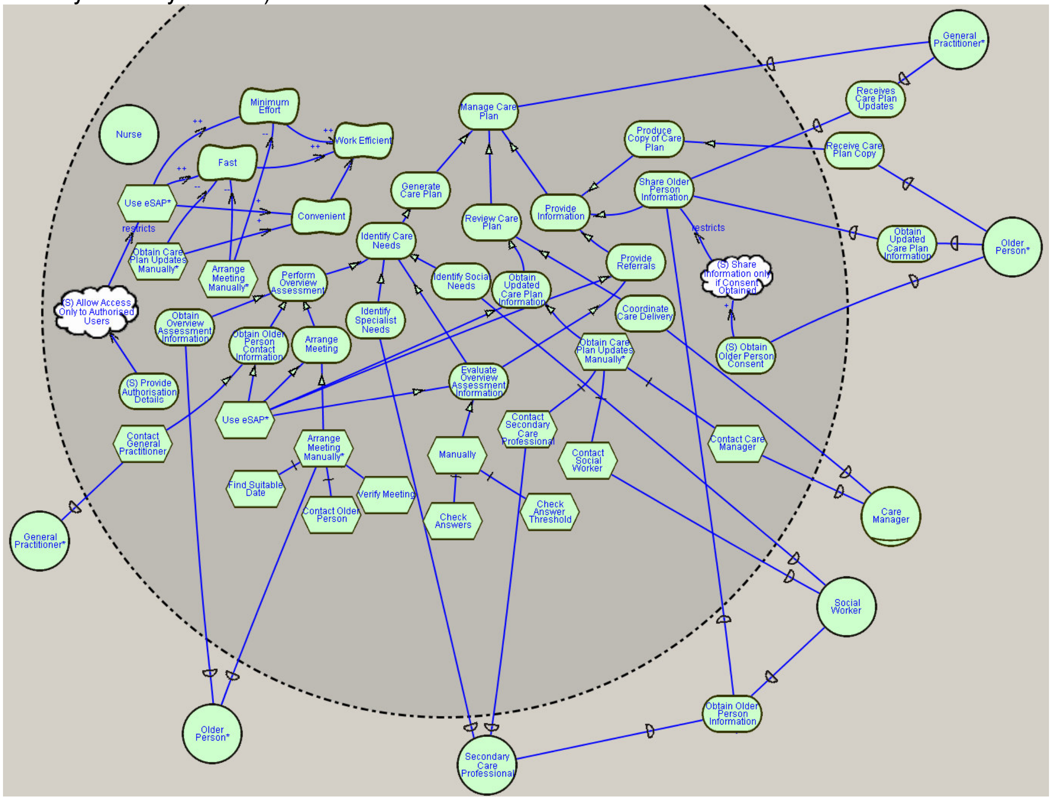

General Practitioner*
Receives Care Plan Updates
Receive Care Plan Copy
Nurse
Minimum Effort
Work Efficient
Fast
Manage Care Plan
Produce Copy of Care Plan
Share Older Person Information
Use eSAP*
restricts
Convenient
Generate Care Plan
Review Care Plan
Provide Information
restricts
Obtain Updated Care Plan Information
Older Person*
Obtain Care Plan Updates Manually*
Arrange Meeting Manually*
Perform Overview Assessment
Identify Care Needs
Identify Social Needs
Obtain Updated Care Plan Information
Provide Referrals
(S) Share Information only if Consent Obtained
(S) Allow Access Only to Authorised Users
Obtain Overview Assessment Information
Obtain Older Person Contact Information
Arrange Meeting
Identify Specialist Needs
Coordinate Care Delivery
Obtain Care Plan Updates Manually*
(S) Obtain Older Person Consent
(S) Provide Authorisation Details
Evaluate Overview Assessment Information
Use eSAP*
Contact General Practitioner
Arrange Meeting Manually*
Contact Secondary Care Professional
Manually
Contact Social Worker
Contact Care Manager
Find Suitable Date
Verify Meeting
Contact Older Person
Check Answers
Check Answer Threshold
Care Manager
General Practitioner*
Social Worker
Older Person*
Secondary Care Professional
Obtain Older Person Information

Figure 3: Partial Goal Diagram for the Nurse Actor

Therefore, the security constraint Allow Access Only to Authorised Users, which restricts the Use eSAP task, is imposed to the Nurse actor. To help towards the satisfaction of the imposed security constraint the secure goal Provide Authorisation Details is introduced to the Nurse as indicated in Figure 3.
Modelling the actors internally leads to a more precise definition of the why of the system functionality, and this subsequently helps to verify how the final system implementation matches the stakeholders' real needs.
When all the actors have been further analysed, the actor diagram is refined, as shown in Figure 4, and any possible new dependencies identified during the internal actors' analysis are modelled.

[5] To keep the complexity of the figure as minimum as possible, an asterisk * has been used to indicate that the same actor, goal, or entity has been modelled more than once in the figure.

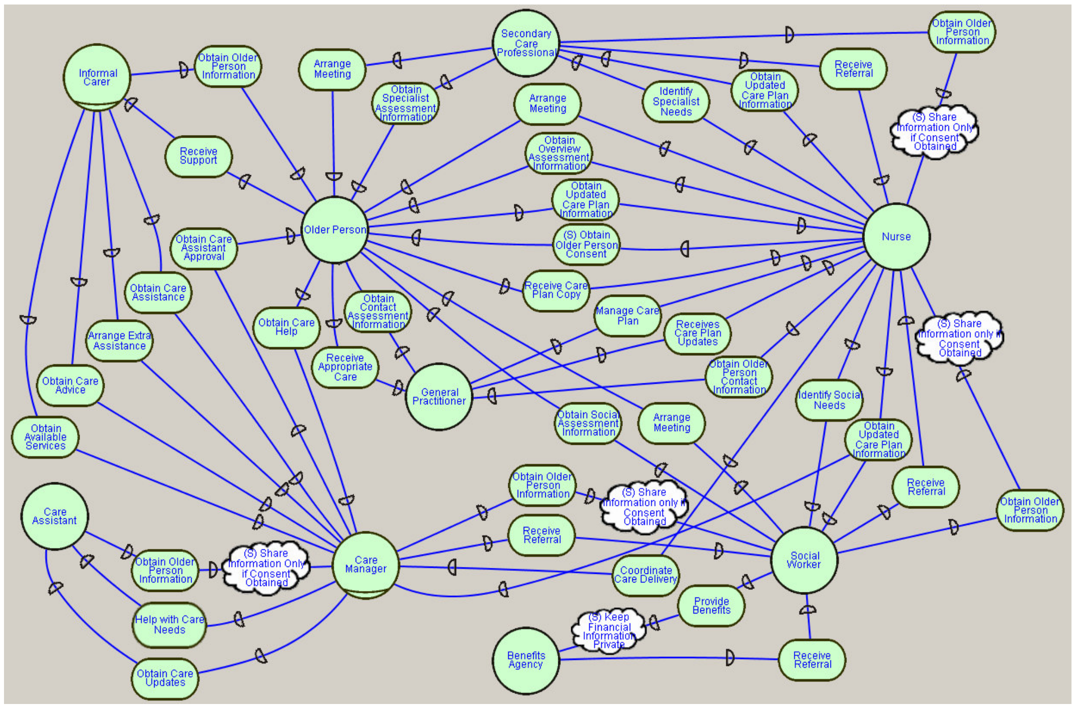


Figure 4: Refined Actor Diagram

This is important since during the actors' internal analysis it is possible that new goals are discovered, which the actors might not be able to satisfy by themselves. Therefore, new dependencies are introduced to enable an actor to delegate to another actor the goals that cannot accomplish on their own. Introducing new dependencies between the actors according to the goals/tasks derived from the internal analysis that takes place in each actor is a common task in Secure Tropos and it is very important since it helps to identify clearly the relationships between the actors and also indicates how the analysis of the goals of one actor can influence the dependencies between this actor and the other actors. Refining the dependencies and the social relationships of the actors this way, leads to a more precise definition of the why of the system functionalities, and as a last result, helps to verify how the final implementation matches the real needs [17]. Also, with regards to security the refinement of the actor diagram is equally important. Refining an actor's goals and dependencies could result in the redefinition of the security constraints imposed to particular dependencies or the addition of new security constraints. As an example, consider the security constraint Share Information Only if Consent Obtained. This security constraint was imposed to the Nurse, as shown in Figure 2, by the Older Person as part of the Obtain Overview Assessment Information dependency. However, the internal analysis of the Nurse indicated that this security constraint restricts in fact the Share Older Person Information goal of the Nurse. Therefore, in the refined actor diagram, the security constraint has been imposed to all the newly discovered (after the internal analysis of the actors) dependencies that involve the Share Older Person Information goal.
One of the most important issues in the development of the electronic Single Assessment Process system, as in any health and social care information system, is to precisely identify the environment that the system will be situated. This means to identify the intentions of the users, anyone related to the system, and the relationships between those. This is necessary in order to identify the functional and non-functional requirements of the system-to-be. Secure Tropos helps towards this direction during the early requirements analysis stage, since the final output of the early requirements analysis is a clearly defined organisational model, which includes relevant actors (users), their intentions and their respective dependencies (relationships).

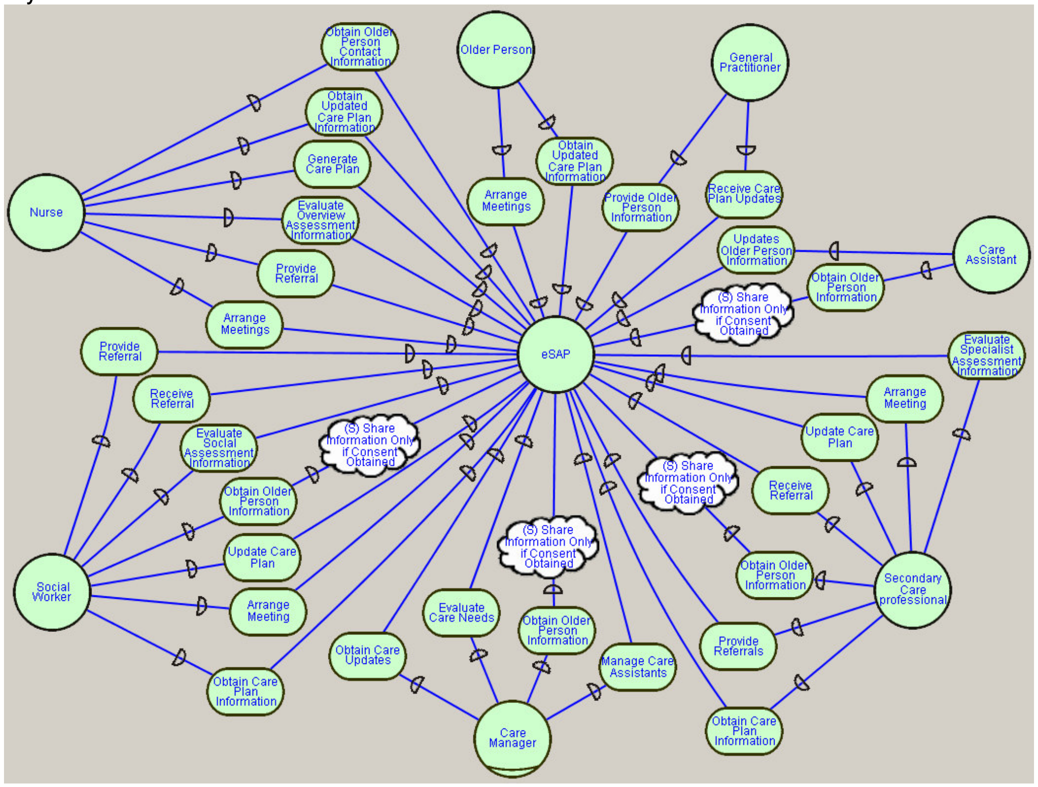
6.2.2 Late requirements analysis

During the late requirements analysis the system-to-be, in our case study the electronic Single Assessment Process system, is described within its operation environment along with relevant functions and qualities. The system is presented as one or more actors, who have a number of dependencies with the other actors of the organisation. These dependencies define all functional and non-functional requirements for the system-to-be.

During the Nurse analysis, modelled in Figure 3, it was identified that the Nurse can achieve some of the goals either manually or by using the electronic single assessment process (eSAP) system. Consider for example, the Arrange Meeting goal of the Nurse actor. This can be fulfilled either by the task Use eSAP or by the task Arrange Meeting Manually. However, the analysis, presented in Figure 3, showed that by using an information system, the eSAP system, the Nurse will be able to work more efficiently, with less effort, faster and more conveniently than trying to achieve the task manually. Similar conclusions were drawn for all the actors of the system [14]. Therefore, it was decided that the use of eSAP provides advantages over the manual achievement of most of the actors' tasks, and as a result the responsibility for the achievement of those tasks was delegated to the eSAP system.

Therefore, in our analysis, the eSAP system has been introduced as another actor that receives the responsibility for the fulfilment of some of the goals identified during the early requirements analysis for the actors of the system. In other words, some goals that the actors of the system cannot fulfil or are better fulfilled by the eSAP system are delegated to the eSAP System.

The actor diagram including the eSAP system and the refined dependencies6 is shown in Figure 5. For instance, it was identified during the early requirements analysis that the Arrange Meetings and Obtain Updated Care Plan Information goals of the Older Person will be achieved better with the aid of the eSAP system.


Obtain Older Person Contact Information
Older Person
General Practitioner
Obtain Updated Care Plan Information
Generate Care Plan
Nurse
Evaluate Overview Assessment Information
Arrange Meetings
Obtain Updated Care Plan Information
Provide Older Person Information
Receive Care Plan Updates
Updates Older Person Information
Care Assistant
Obtain Older Person Information
Provide Referral
(S) Share Information Only if Consent Obtained
Arrange Meetings
Provide Referral
eSAP
Evaluate Specialist Assessment Information
Receive Referral
Arrange Meeting
Evaluate Social Assessment Information
(S) Share Information Only if Consent Obtained
Update Care Plan
(S) Share Information Only if Consent Obtained
Receive Referral
Obtain Older Person Information
Update Care Plan
(S) Share Information Only if Consent Obtained
Secondary Care professional
Social Worker
Arrange Meeting
Evaluate Care Needs
Obtain Older Person Information
Obtain Older Person Information
Provide Referrals
Obtain Care Updates
Manage Care Assistants
Obtain Care Plan Information
Care Manager
Obtain Care Plan Information

[6] It is worth mentioning that the dependencies of the Informal Carer actor are not delegated to the eSAP system, since it is assumed at this point that the Informal Carer does not interact with the system.

Figure 5: Actor diagram including the eSAP actor

Therefore to indicate this, two goal dependencies have been added from the Older Person to the eSAP system actors. It is worth mentioning that along with dependencies, security constraints are also delegated. For example, before the introduction of the eSAP system, the Social Worker was depending on the Nurse to Obtain Older Person Information. However, this secure dependency involves the security constraint (restricting the Nurse) Share Information Only if Consent Obtained. With the introduction of the eSAP system, the Social Worker actor depends on the eSAP for the satisfaction of the Obtain Older Person Information secure dependency. Therefore, the satisfaction for this dependency is delegated from the Social Worker to the eSAP actor. However, this secure dependency imposes the satisfaction of the Share Information Only if Consent Obtained security constraint, and as a result the eSAP becomes responsible for satisfying it.
To satisfy all the delegated dependencies, the main goal of the eSAP system has been identified as to Automate Care. By performing a means-end analysis, presented in Figure 6, it was identified that for the eSAP System to fulfil the Automate Care goal, the following sub-goals must be accomplished: Assist with Assessment Procedures, Provide Older Person Information, Manage Care Plans and Schedule Meetings. Each of these sub-goals can be furthered analysed by employing means-end analysis. For example, the Manage Care Plans goal can be accomplished with the fulfilment of the Generate Care Plan, Manage Care Plan Updates, Provide Care Plan Information, Manage Referrals and Identify Care Assistants sub-goals.

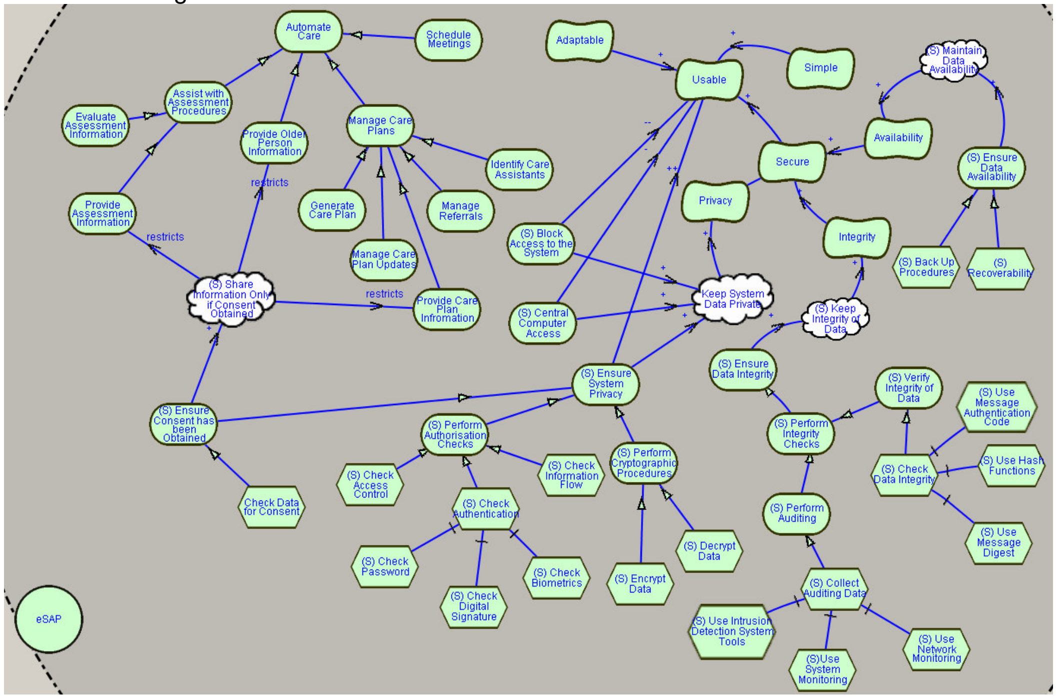

Automate Care
Schedule Meetings
Adaptable
Usable
Simple
(S) Maintain Data Availability
Evaluate Assessment Information
Assist with Assessment Procedures
Provide Older Person Information
Manage Care Plans
Identify Care Assistants
Availability
Secure
(S) Ensure Data Availability
Provide Assessment Information
restricts
restricts
Generate Care Plan
Manage Referrals
Privacy
Integrity
(S) Block Access to the System
Manage Care Plan Updates
(S) Back Up Procedures
(S) Recoverability
(S) Share Information Only if Consent Obtained
restricts
Provide Care Plan Infromation
Keep System Data Private
(S) Central Computer Access
(S) Keep Integrity of Data
(S) Ensure Data Integrity
(S) Ensure System Privacy
(S) Verify Integrity of Data
(S) Use Message Authentication Code
(S) Ensure Consent has been Obtained
(S) Perform Authorisation Checks
(S) Perform Integrity Checks
(S) Perform Cryptographic Procedures
(S) Check Information Flow
(S) Check Access Control
(S) Check Data Integrity
(S) Use Hash Functions
Check Data for Consent
(S) Check Authentication
(S) Perform Auditing
(S) Check Password
(S) Decrypt Data
(S) Use Message Digest
(S) Check Biometrics
(S) Encrypt Data
(S) Collect Auditing Data
eSAP
(S) Check Digital Signature
(S) Use Intrusion Detection System Tools
(S)Use System Monitoring
(S) Use Network Monitoring

Figure 6: Goal Diagram for the eSAP actor

From the security point of view, three main constraints imposed, by the desired security features of the system, Privacy, Integrity and Availability, to the eSAP's main goal. These are Keep System Data Private, Keep Integrity of the Data and Maintain Data Availability. In addition, the eSAP system must satisfy the Share Information Only if Consent Obtained security constraint imposed to the eSAP by the secure dependencies delegated by the other actors. Each of these secure constraints can be satisfied with the aid of one or more secure goals. For example, the Keep System Data Private security constraint can be fulfilled by blocking access to the system, by allowing access only from a central computer, or by ensuring system privacy. However, the first two contribute negatively to the usability of the system, i.e. the system will be secure but it will not be used. On the other hand, the Ensure System Privacy secure goal is considered the best solution since it provides security to the system and it doesn't affect (dramatically) its usability.

Thus, for the eSAP to satisfy its security constraints the following secure goals have been identified as shown in Figure 6: Ensure System Privacy, Ensure Data Integrity, Ensure Data Availability and Ensure Consent has been Obtained. These can be furthered analysed. For example, the Ensure System Privacy goal is further analysed into the Perform Authorisation Checks and Perform Cryptographic Procedures secure goals. Both of those goals must be fulfilled for the Ensure System Privacy goal to be satisfied.

An important issue at this point is to check whether the goals assigned to the eSAP system satisfy all the goals delegated to the system by the other actors. Thirty (30) goals were delegated to the eSAP system as shown in Figure 5. From these goals, fifteen of them are satisfied by the Manage Care Plans goal (and its sub-goals), six of them are satisfied by the Provide Older Person Information goal, five are satisfied by the Assist with Assessment Procedures goal (and its sub-goals), and four of them are satisfied by the Schedule Meetings goal.

As it can be seen from the analysis presented in this section, the late requirements analysis stage follows the same analysis techniques used in the early requirements analysis. The main difference is the idea of introducing the system as another actor. Such an approach is very important and provides advantages since it helps to identify clearly the relationships and the dependencies between the system and the environment that the system will be situated. Medical Information Systems, such as the electronic Single Assessment Process, more often are introduced to environments in which non or very little computer expertise is found. Defining clearly the roles and the dependencies of the actors and the system helps to identify the functional and non-functional requirements of the system-to-be according to the real needs of the stakeholders. Also, by analysing the system within its operational environment helps to delegate responsibility for the achievement of goals to the system (by other stakeholders) and also identify new dependencies between the system and its stakeholders. This leads to the definition of functional and non-functional requirements for the system, which would be very difficult to identify otherwise. In addition, the way the system is analysed within the late requirements stage, provides developers with the ability to consider different alternatives for satisfying the system's goals and decide, by checking for example if the alternative contributes positively or negatively to other goals of the system, which of those alternatives is the best solution.

6.2.3 **Architectural design**

When the system goals have been identified, the next step of the development cycle involves the definition of the system's global architecture in terms of subsystems (actors) interconnected through data and control flows (dependencies). To achieve this, Secure Tropos employs different processes during the architectural design stage, such as the Addition of new actors, in which new actors (and sub-actors) are added to make the system interact with the external actors, to take responsibility to fulfil one or more goals of the system and to contribute positively to the fulfilment of some non-functional requirements; the Capabilities identification, in which the capabilities needed by the actors to fulfil their goals and tasks are identified; and the Agents assignment, in which a set of agent types is defined assigning to each agent one or more different capabilities.

The first step on the architectural design is the choice of the architectural style to be used for the system. For the eSAP system a client/server architectural style has been chosen [17], as opposed to a mobile agents style. Mainly, this decision took place because the client/server style contributes more to the security of the eSAP system than the mobile agents style [17].

When the architectural style has been chosen, the next step of the architectural design stage aims to decompose the system in order to identify internal actors who will satisfy the system's goals. In the presented example, the eSAP actor is decomposed, as shown in Figure 7, to internal actors and the responsibility for the fulfilment of the eSAP's goals is delegated to these actors.

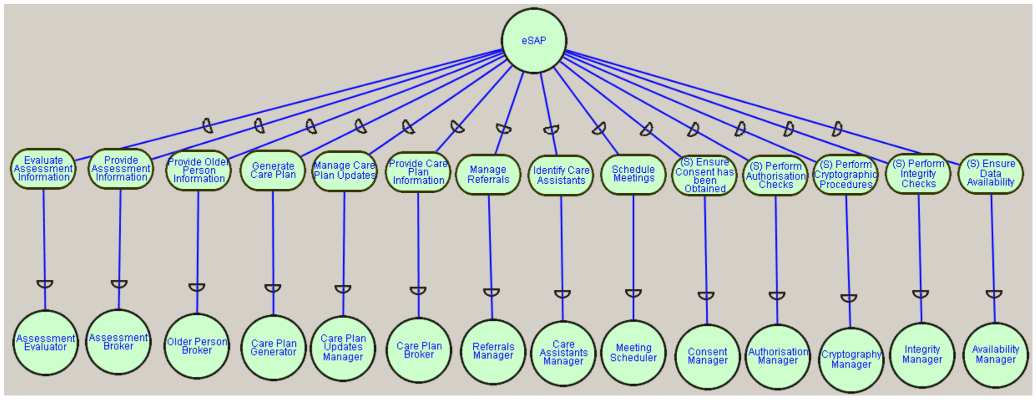


Figure 7: Partial Decomposition of the eSAP system

For instance, the Evaluate Assessment Information goal is delegated to the Assessment Evaluator, whereas the Provide Assessment Information goal is delegated to the Assessment Broker. In addition, the Older Person Broker and the Consent Manager actors have been introduced to the eSAP system to fulfil the responsibility (identified during the late requirements analysis – see Figure 5)of the eSAP system to satisfy the secure dependency Obtain Older Person Information together with the Share Information Only if Consent Obtained security constraint.

However, the new introduced actors must be furthered analysed and their dependencies with the other (existing and new) actors must be furthered investigated. Such an analysis is important since it helps developers to identify dependencies between new and existing actors, introduce new actors to the system-to-be and, as a result of this, refine the goals of the system or even possibly introduce new goals to the system, which would be very hard to identify otherwise.

An important point at this stage of the development is the identification of the actors to deliver the security goals of the system. This is important because, as opposed to the identification of actors to deliver the functional goals of the system, the identification of the "security" actors is a difficult task, especially for developers with minimum knowledge of security (this is the case for most of the information system developers). To help developers with this task, Secure Tropos employs a security pattern language [17]. Security patterns can greatly help to identify the required actors in a structured manner, which does not endanger the security of the system, by providing a solution customised to the problem.

For example, from the internal analysis of the eSAP, presented in Figure 6, it was concluded that Information Flow, Authentication and Access Control checks must be performed in order for the eSAP system to satisfy the secure goal Ensure System Privacy. In the case of the information flow secure task, the eSAP should be able to control how information flows within the system, and between the system and other actors. For example, the system should be able to control who requires access to the system and, by considering the security policy, to grant or deny access to the system. With respect to the Authentication checks, the system should be able to authenticate any agents that send a request to access information of the system, and in the case of the access control, the system should be able to control access to its resources.

The pattern language[7] proposed in [17] can be used to fulfil the above-mentioned secure goals of the eSAP system. Three main patterns are employed by the language: The Agency Guard, which provides a single non-bypassable point of access, the Agent Authenticator which provides authentication services to an agency, and the Access Controller, which provides access to resources according to a security policy. Therefore, in the case of the eSAP, the Agency Guard pattern can be used to check grant/deny access to the system according to the security policy, the Agent Authenticator pattern can be used to provide authentication checks and the Access Controller pattern to perform access control checks. The use of these patterns not only satisfies the fulfilment of the secure goals of the system but also guarantees the validity of the solution.

To apply a pattern, the developer must carefully consider the problem to be solved and the consequences that the application of each particular pattern will have on the system. Figure 8 illustrates a possible use of the Agency Guard, Agent Authenticator and Access Controller patterns in the eSAP system.

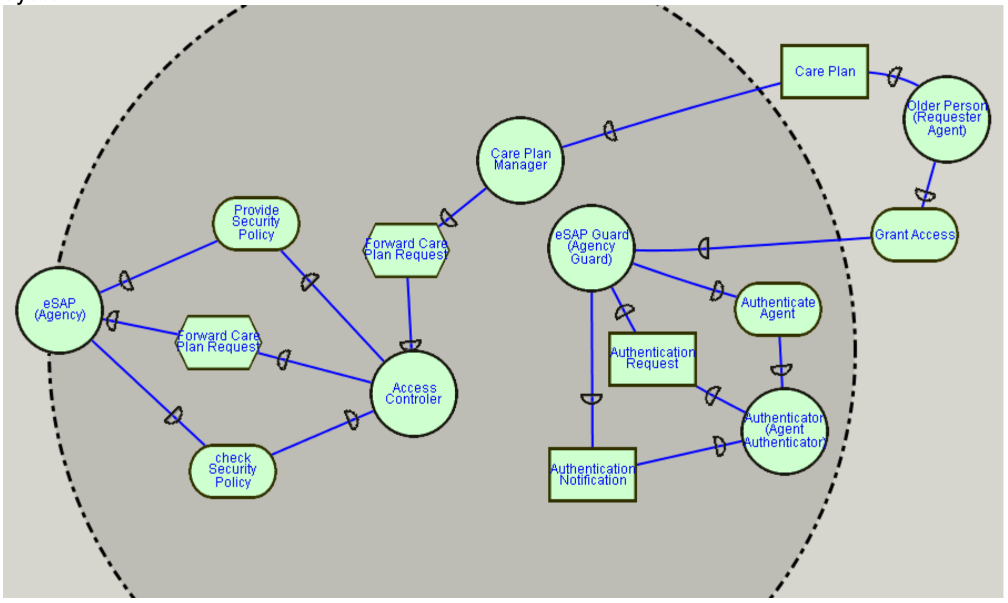

Care Plan
Older Person (Requester Agent)
Care Plan Manager
Provide Security Policy
Forward Care Plan Request
eSAP Guard (Agency Guard)
Grant Access
eSAP (Agency)
Authenticate Agent
Forward Care Plan Request
Authentication Request
Access Controler
Authenticator (Agent Authenticator)
check Security Policy
Authentication Notification

Figure 8: Using the AGENCY GUARD, the AGENT AUTHENTICATOR and the ACCESS CONTROLLER patterns in the development of the eSAP

In particular it shows how the secure goals Check Information Flow (problem), Check Authentication (problem) and Check Access Control (problem) can be satisfied. The Agency Guard satisfies the goal by providing a single non-bypassable point of access to the system (solution), the Agent Authenticator satisfies the goal by authenticating each agent that tries to access the system (solution) and the Access Controller controls access to the resources of the system (solution). The use of the patterns helps developers to delegate the responsibilities of particular system security goals to particular actors defined by the patterns. In addition, the developer knows the consequences that each pattern introduces to the eSAP system.

[7] Although different patterns might be employed, the chosen language has been developed with agent orientation in mind –as opposed to many object oriented languages- and as such it provides concepts similar to the Secure Tropos concepts. This results in a unified modelling process.

The application of the Agency Guard means that only the Agency Guard must be tested for correct enforcement of the agency’s security policy (consequence), the application of the Agent Authenticator means that during implementation only the Agent Authenticator must be checked for assurance (consequence), whereas the application of the Access Controller means that different policies can be used for accessing different resources (consequence).
Therefore, as derived from the application of the pattern language, the eSAP delegates responsibility for the fulfilment of the Perform Authorisation Checks security goal to three new actors, the eSAP Guard (delegated the Check Information Flow secure task), the Authenticator (delegated the Check Authentication secure task), and the Access Controller (delegated the Check Access Control secure task) as shown in Figure 9.

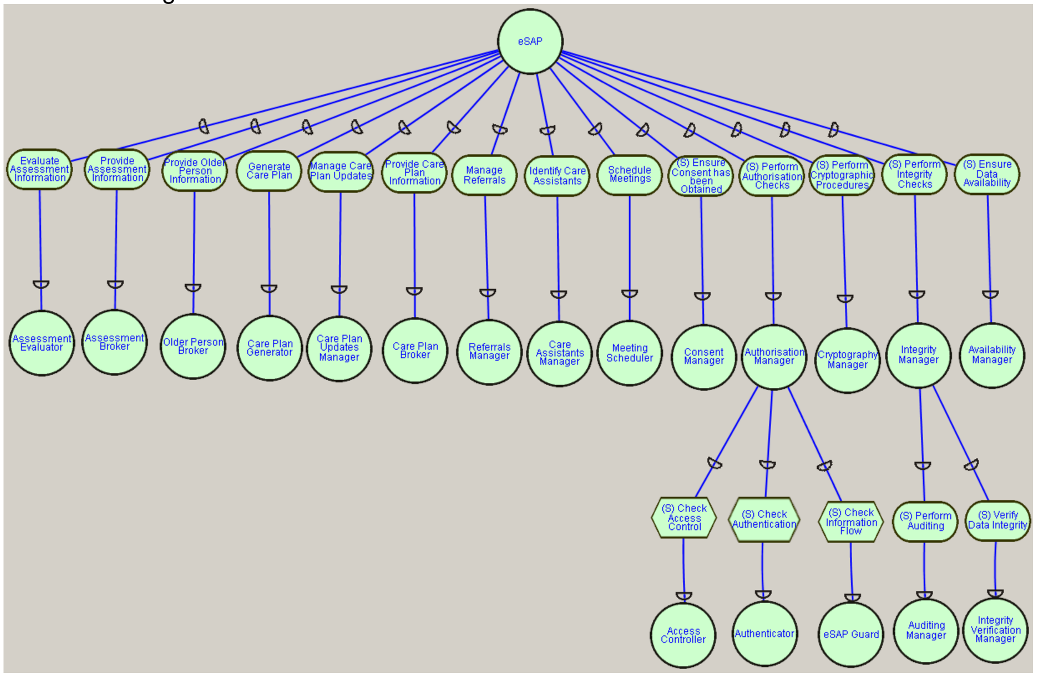


Figure 9: Decomposition of the authorisation and integrity managers

In addition, the Secure Tropos methodology introduces extended actor diagrams, in which the new actors and their dependencies with the other actors are presented. As an example, consider the extended diagram depicted in Figure 10.

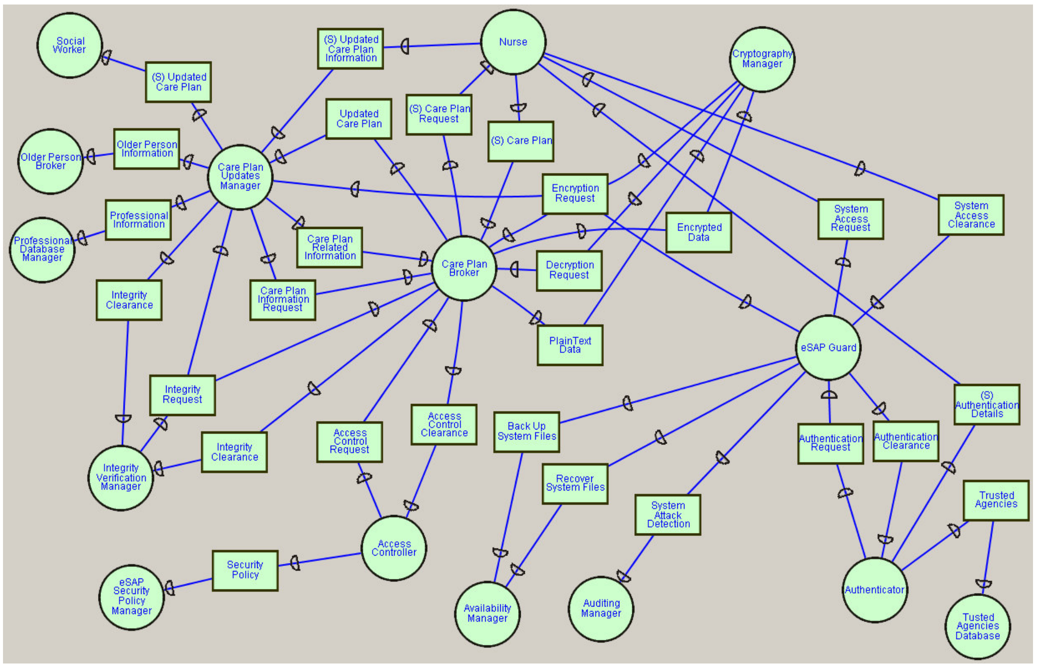


Figure 10: Extended diagram for the eSAP

In this diagram8 the resource dependencies between the Social Worker, the Older Person Broker, the Care Plan Updates Manager, the Nurse, the Cryptography Manager, the Care Plan Broker, the eSAP Guard, the Access Controller, the Availability Manager, the Auditing Manager, the Integrity Verification Manager, and the Authenticator are modelled. An important point to consider is the addition of new actors, such as the Professional Database Manager, the eSAP Security Policy Manager, and the Trusted Agencies Database as derived from the analysis of the other actors in order to fulfil the delivery of specific resources such as the Professional Information, or the system's security policy.
In addition, the extended diagram can be further analysed in order to model more precisely the actors. Consider for instance, the extended diagram with respect to the Assessment Evaluator actor, as depicted in Figure 11. The Assessment Evaluator has been delegated the responsibility to satisfy the goal Evaluate Assessment Information. To fulfil this goal, the Assessment Evaluator depends on two internal actors, the Assessment Analyser and the Evaluation Synthesiser. The first is responsible for obtaining the Assessment Information secure resource, identify the problems of the Older Person according to the Assessment Information and provide the Problems to the Evaluation Synthesiser. The latter is responsible for obtaining the Evaluation Request, and the Problems and providing the Assessment Evaluation secure resource to the actor requesting the information (in the presented analysis to the Social Worker) after considering the Problems, the Available Professionals, the Required Skills and the Proposed Actions resources.
In addition, at this stage, the capabilities identification, in which the capabilities needed by each actor to fulfil their goals and tasks are modelled. Each actor's capabilities can be identified with the aid of the extended actor diagram, since each resource dependency relationship can give place to one or more capabilities triggered by external events. For example the resource Evaluation Request, shown in Figure 11, calls for the capability Obtain Evaluation Request for the Evaluation Synthesiser actor and Provide Evaluation Request for the Social Worker actor.

[8] In order to keep the diagram simple, only some of the actors of the eSAP system have been included in this diagram.

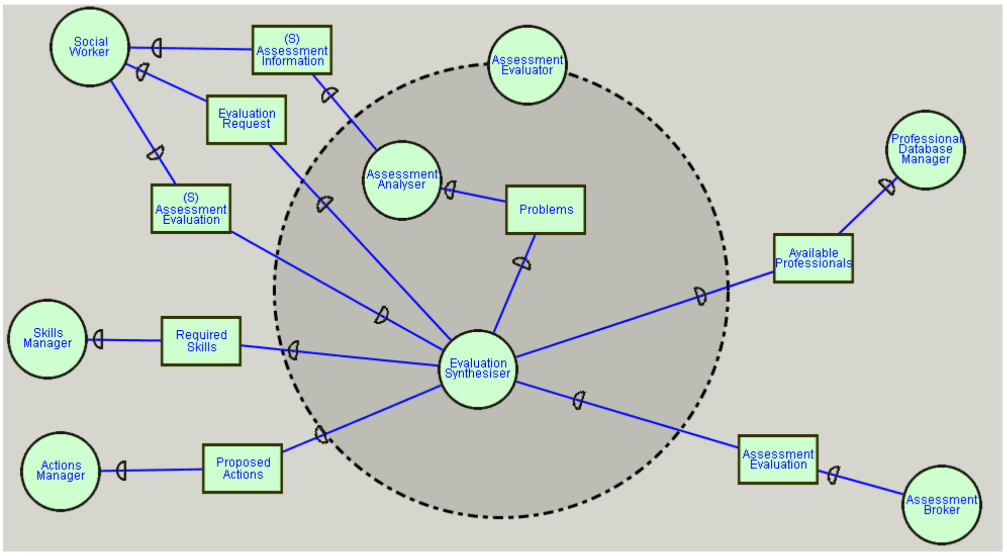


Figure 11: Extended actor diagram with respect to the Assessment Evaluator

In addition, capabilities are identified taking into account the resources of the extended actor diagram. For example, as identified in the early requirements analysis, for the eSAP system to satisfy the Ensure System Privacy secure goal, only encrypted data transfers across the network should be allowed. Therefore, the Assessment Information resource sent from the Social Worker to the Assessment Analyser must be encrypted. Because of this the Social Worker actor should be provided with capabilities to encrypt and decrypt data. Later in the detailed design, each agent's capabilities are further specified and then coded during the implementation phase. Table 1 reports the actors of Figure 11 and their capabilities as derived from the dependencies that exist between them.
When all the actors and their capabilities have been identified, the next step of the architectural design is the agents' assignment. During this step a set of agents are defined and each agent is assigned one or more different capabilities, as shown in Table 2. The capabilities are assigned according to the actors that the agent represents.

Table 1: Actors and their capabilities with respect to the extended diagram of Figure 11

Actor	Capability	Capability Id.
Assessment Analyser	Get Assessment Information	1
	Provide Problems	2
Evaluation synthesizer	Get Problems	3
	Get Evaluation Request	4
	Provide Assessment Evaluation	5
	Get Required Skills	6
	Get Available Professionals	7

	Get Proposed Actions	8
Skills Manager	Provide Required Skills	9
Professional Database Manager	Provide Available Professionals	10
Actions Manager	Provide Proposed Actions	11
Assessment Broker	Get Assessment Evaluation	12
Social Worker	Provide Assessment Information	13
	Provide Evaluation Request	14
	Get Assessment Evaluation	15
	Encrypt Data	16
	Decrypt Data	17

Table 2: Agent types and their capabilities

Agent	Capabilities
Assessment Evaluator	1,2,3,4,5,6,7,8
Skills Manager	9
Professional Database Manager	10
Actions Manager	11
Assessment Broker	12
Social Worker	13,14,15,16,17

Detailed design

The next step of the development process involves the specification of the system's components.
Detailed design stage aims at specifying agent capabilities, plans, and interactions and it is intended to introduce additional detail for each architectural component of the system. In Secure Tropos the detailed design stage is based on the specifications resulting from the architectural design phase and the reasons for a given element, designed at this level, can be traced back to early requirements analysis, a very important advantage of the methodology. Another important advantage of the methodology is that it is well integrated with existing work. Thus, for the detailed design stage, Secure Tropos adapts a subset of the Agent Unified Modelling Language (AUML) diagrams proposed in [23]. These are:
Capability Diagrams. We use AUML activity diagrams to model a capability or a set of capabilities for a specific actor. In each capability diagram, the starting state is represented by external events, activity nodes model plans, transition arcs model events, and beliefs are modelled as objects. Consider for example, the Receive Care Plan Request secure capability of the Care Plan Broker. This can be depicted as shown in Figure 12.

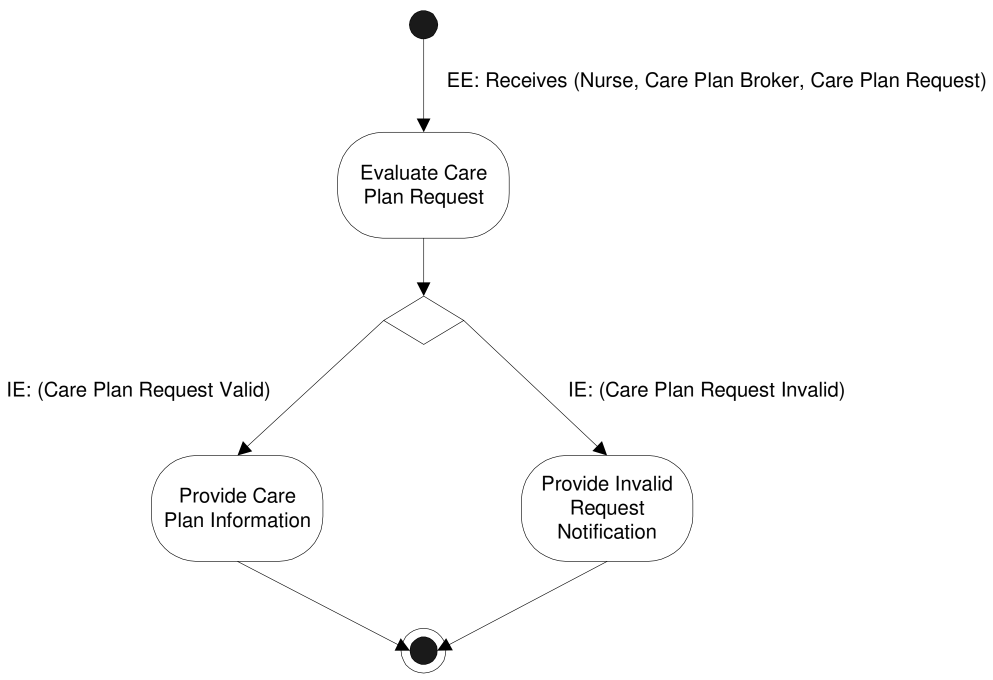

EE: Receives (Nurse, Care Plan Broker, Care Plan Request)
Evaluate Care Plan Request
IE: (Care Plan Request Valid)
IE: (Care Plan Request Invalid)
Provide Care Plan Information
Provide Invalid Request Notification

Figure 12: Capability diagram for the Receive Care Plan Request capability of the Care Plan Broker **agent**

The Care Plan Broker initially accepts a Care Plan Request from the Nurse. Then the Care Plan Broker evaluates the request and either provides the requested information or notifies the requester (the Nurse in this case) that the request is invalid.

Plan Diagrams. Plan Diagrams are used to further specify each plan node of a capability diagram. The starting point of a plan diagram is the initiation of a plan and the end point is the termination of a plan. The different actions required by the plan are modelled as activity nodes together with the transitions (modelled as arcs) from one actions to a subsequent one. For instance, the Evaluate Care Plan Request plan depicted in Figure 12, is modelled as shown in Figure 13. The Care Plan Broker firstly tries to decrypt the incoming request. If the request is not encrypted then the agent categorises the request as not valid (all the incoming requests must be encrypted) and the plan is terminated. If the request is successfully decrypted the next step involves the integrity check of the request. In case the integrity of the request is not verified the agent categorises the request as not valid and the plan is terminated. The last step involves reading the request in order for the agent to respond to it. It must be noticed that every incoming request follows a specific format, in order for the agent to be able to read it. If the request is readable the Care Plan Broker categorises it as valid request and the plan terminates, in any other case the request is categorised as invalid and the plan terminates.

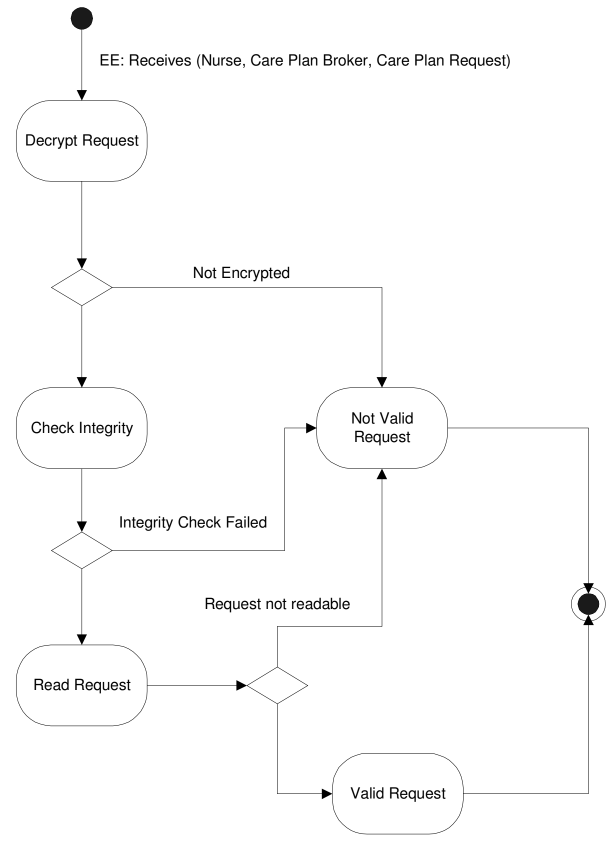


Figure 13: Plan diagram for the evaluate care plan request plan

Agent Interaction Diagrams. We apply in our case sequence diagrams modelling agent Interaction Protocols as proposed by [25]. Agent Interaction Diagrams capture the structural patterns of interactions between the agents of the system by emphasising the chronological sequence of communications. Consider for example, the interactions that take place when the Social Worker tries to obtain access to the system as shown in Figure 14. The Social Worker sends an encrypted message to the eSAP Guard requesting access to the system. The eSAP Guard forwards the request to the Cryptography Manager for decryption. After the Cryptography Manager decrypts the request it forwards it plain text to the eSAP Guard. Then the eSAP Guard checks the authentication privileges of the Social Worker with the aid of the Authenticator. Then the Authenticator requests from the Social Worker to send their authentication details. When the Authenticator receives the authentication details of the Social Worker either provides an authentication clearance or rejects the authentication of the Social Worker. After the authentication clearance has been granted, the eSAP Guard provides system access clearance to the Social Worker.

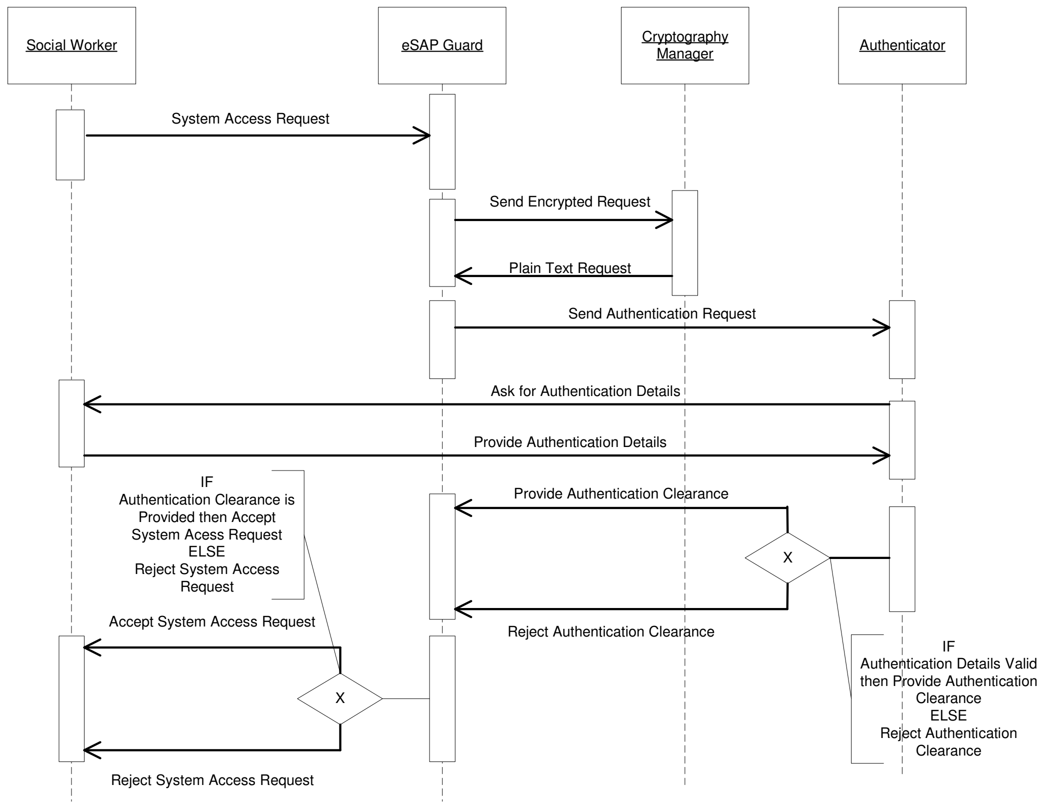

Social Worker
eSAP Guard
Cryptography Manager
Authenticator
System Access Request
Send Encrypted Request
Plain Text Request
Send Authentication Request
Ask for Authentication Details
Provide Authentication Details
IF Authentication Clearance is Provided then Accept System Acess Request ELSE Reject System Access Request
Provide Authentication Clearance
X
Reject Authentication Clearance
Accept System Access Request
X
IF Authentication Details Valid then Provide Authentication Clearance ELSE Reject Authentication Clearance
Reject System Access Request

Figure 14: Interaction diagram for the Social Worker system access clearance

The detailed design stage takes place until all the components of the system identified during the architectural design have been specified in depth with the aid of the methods and diagrams presented in this section.

Discussion regarding the case study

As a result of the application of Secure Tropos to the above case study, we are able to draw some important conclusions regarding the degree in which Secure Tropos satisfies the requirements identified in Section 0.

Requirement 1*:*

An appropriate methodology should provide concepts and procedures to decompose effectively the system to smaller, easier to understand and develop components.

The concepts that Secure Tropos employs throughout the development stages, allow developers to decompose easily the system to smaller components by mapping the system development concepts to entities of the real environment in which the system will be situated. For example, initially the system was modelled as one actor. By analysing the goals of the system, and assigning sub-actors to satisfy these goals, the decomposition of the system becomes easier. Moreover, the developers can verify whether the decomposed system meets the initial requirements by analysing if the new actors meet the delegated goals. In addition, the application of patterns during the architectural design of the Tropos methodology provides developers with an additional "tool" to help them to decompose the system effectively. On the other hand, Secure Tropos defines three different processes (Addition of new actors, Capabilities identification and agent assignment – for a reminder please see Section 6.4) to allow developers with less experience in decomposing a system to successfully achieve decomposition.

Requirement 2:

An appropriate methodology should allow developers to iterate during the development stages and easily modify/add/delete the various components of the system.

Although, for the sake of simplicity we presented the stages and processes of the Secure Tropos methodology in a sequential way, the development process is actually an iterative one, in which developers and users can move back and forward on the development to add/modify different components that might be needed by the system. For example, during the development of the eSAP case study, initially the goals of the care manager role had been assigned to the Social Worker actor. However, as it was concluded after discussions and input from the users (health and social care professionals), this is not always the case. It is possible a health professional will assume responsibility for the satisfaction of these goals. Therefore, we modified the actor diagram to include the role of the care manager, which in our case study is played by the Social Worker.

Requirement 3*:*
An appropriate methodology should provide concepts that are easily understood not only by the system developers but also from the system's stakeholders/users.
Although, we did not contact a detailed survey as to whether the concepts of the methodology are easy to understand and use by health and social care professionals, their involvement throughout the development process indicated they were feeling confident with them. After an initial "learning" period the users of the system (mainly health professionals) started to provide feedback about the system using the same concepts and notations as the developers. This we believe, made the analysis and design faster and helped greatly to establish a mutual understanding.

Requirement 4:
An appropriate methodology should allow developers to model not only the system but also the environment in which the system will be deployed as well as the relationships between the different stakeholders/users of the system.
One of the novelties of the Secure Tropos methodology is that it starts the development process from the early requirements analysis, in which the environment of the system is modelled. Regarding the eSAP case study, neglecting any reference to an electronic system during the early requirements phase allowed us to precisely identify the environment of the Single Assessment Process, and moreover identify the reasons for introducing an electronic system (see for example the Goal Diagram for the Nurse actor in Figure 3). In addition, the usage of concepts such as goals and dependencies allowed us to precisely identify all the relationships and the delegations of goals between the actors of the case study.

Requirement 5:
An appropriate methodology should allow developers to consider security issues throughout the development stages by providing concepts and processes customised to security.
By modelling the security constraints of the individual actors, developers are able to model the security requirements of the system according to the real security needs of its stakeholders. For example, during the eSAP system analysis, the lack of identifying the security constraints between the Nurse and the Older Person, or the Social Worker and the Benefits Agency would result in a design that would miss important information regarding the security of the system. Furthermore, by imposing security constraints, and differentiate between security-related and non-security-related goals and entities, developers can define together security and other functional and non-functional requirements of a system and at the same time provide a clear distinction between them. This distinction helps towards the detection of possible conflicts between security and other requirements, and therefore allows developers to analyse those conflicts and propose possible ways towards a design that will overcome them, leading to the development of a more secure system.

Related Work

This paper proposes the Secure Tropos agent oriented software engineering methodology as a suitable methodology for the development of health and social care information systems.
Most of the methodologies used so far for the development of health and social care information systems are based on the object oriented software engineering paradigm [5], according to which an information system is decomposed to smaller components which communicate in terms of messages. The object orientation paradigm uses concepts such as object, classes, attributes, and methods, which are well known amongst the system developers. Beer et al. [26] employ object orientation techniques such as use case diagrams and collaboration diagrams to develop an information system for managing

the provision of care. Lee and Asllani [27] propose the SCOPE IT object oriented methodology for designing patient service scheduling systems in health care. Krol and Reich [28] define three models, the object model that describes the hierarchical structure of objects in a system, the dynamic model that represents the sequence of operations of the objects on the system, and the functional diagram to represent the transformation of data within a system by means of data flow diagrams. On the other hand, Blobel [29] argues for the suitability of a component-based approach for the analysis and design of advanced health system architectures.

These approaches, although valuable, demonstrate some important limitations with regard to the development of health and social care information systems. First of all, we feel the detail on which they can model a complex information system, such as a health and social care information system, is restricted. As Booch, one of the "fathers" of object orientation, states [5, page 34] " for complex systems we find that objects, classes and models provide essential yet insufficient means of abstraction". Moreover, the concepts - such as objects, modules, components, and methods - and the modelling languages used by these approaches although very well known and understood within the software engineering community, can be quite confusing for someone not familiar with software engineering methods such as the health and social care professionals. This usually leads to misunderstandings during the crucial stages of the system requirements elicitation and analysis. This misunderstanding might propagate to errors during the design, which are usually cost a large amount of money and time to be corrected when they are discovered, if they are discovered before the implementation. Moreover, current object oriented methodologies fail to adequately consider security during the development of information systems [6], an important limitation, as discussed earlier, when developing information systems for the health and social care sector.

On the other hand, Secure Tropos is not the only agent oriented software engineering methodology. In the last few years many software development methodologies for multi-agent systems have been developed [30], such as MaSE [31], MESSAGE [32], and GAIA [33] amongst others. These approaches start the development process either from the late requirements (system analysis) or the design stages (system design). Such an approach however, is not desirable for the development of health and social care information systems, since the environment and the interactions of the stakeholders play an important – possibly the most important – role when defining the system's functionalities. In contrast, Secure Tropos covers the full range of software development starting from the early requirements (system's environment analysis) and thus allow developers to capture not only the system-to-be but also analyse the environment that the system will be situated and therefore define more precisely the interactions of the system with its stakeholders/ users and as a result the functionalities of the system.

Moreover, unlike other modelling languages used by many agent oriented software engineering methodologies, the Secure Tropos graphical notation and the concepts used in the requirements and architectural design stages (stages that users need to provide feedback and their opinion) are more intuitive and comprehensible for people that are not experts in software engineering [18], such as health and social care professionals. Therefore, the methodology provides an effective means of interaction between the system developers and the users of the system.

Moreover, throughout the paper, we have indicated the necessity to consider security issues during the analysis and design of the electronic Single Assessment Process system. Secure Tropos is the only agent oriented methodology that provides a well defined process for identifying the security requirements of the system, and allow developers to develop a design that satisfies them.

Conclusions and Future Work

The number of health and social care information systems under development increases constantly. An important consideration for such systems is the involvement of the users/stakeholders (health and social care professionals) in the development process. Therefore, it is important that software engineering methodologies exist to assist not only developers but also stakeholders and users. Such methodologies must provide adequate concepts that are understood by the system developers as well as the stakeholders/users of the system, enabling the latter ones to actively involved in the development of the system, and therefore guarantee the correct identification of the system's requirements.

This paper argues for the suitability of the Secure Tropos methodology for the development of health and social care information systems. The argument is mainly based on the degree in which Secure Tropos satisfies the requirements that a methodology for the development of health and social care system should demonstrate as well as on the Secure Tropos key features, such as the use of the same concepts and notations throughout the development process, and the consideration of security as an

integral part of the development process. The applicability of the methodology has been demonstrated by applying it at the electronic Single Assessment Process system case study.
However, this work is not finished. Future work involves the application of the methodology to more health information systems, mainly from different domains, such as cancer information systems, in order to identify whether different health information systems require different modelling techniques or a unique methodology can be used for the development of all health information systems.

References

[1] A. C. Norris, Current Trends and Challenges in Health Informatics, in Proceedings of the 7th International Symposium on Health Information Management Research, Sheffield, June 2002.
[2] C. Dowd, B. Eaglestone, P. Bath, P. Procter (editors), Proceedings of the 7th International Symposium on Health Information Management Research, University of Sheffield, 2002.
[3] Department of Health, Single Assessment Process for Older People, http://www.dh.gov.uk/PolicyAndGuidance/HealthAndSocialCareTopics/SocialCare/SingleAssessmentProcess/fs/en, Last Accessed 11/04/05
[4] N. D. Birrell, M.A. Ould, A practical handbook for software development, Cambridge University Press, 1986
[5] G. Booch, Object-oriented analysis and design – with applications, The Benjamin / Cummings Publishing Company, 1994.
[6] T. Tryfonas, E. Kiountouzis, A. Poulymenakou, Embedding security practices in contemporary information systems development approaches, Information Management & Computer Security, Vol 9 Issue 4,pp 183-197, 2001
[7] L. Liu, E. Yu, J. Mylopoulos, Analyzing Security Requirements as Relationships Among Strategic Actors, in the Proceedings of the 2nd Symposium on Requirements Engineering for Information Security (SREIS'02), Raleigh, North Carolina, October 2002
[8] M. Bradshaw, Software Agents, American Association Artificial Intelligence Publication, 1997
[9] M. Wooldridge, N. R. Jennings, Agent Theories, Architectures, and Languages: A Survey, Intelligent Agents, Wooldridge, Jennings (eds.), Springer-Verlag, pp 1-22, 1995
[10] N. R. Jennings, An agent-based approach for building complex software systems, Communications of the ACM, Vol. 44, No 4, April 2001
[11] M. Wooldridge, P. Ciancarini, Agent-Oriented Software Engineering: The State of the Art, In P. Ciancarini and M. Wooldridge (eds.), Agent-Oriented Software Engineering. Springer-Verlag, Lecture Notes in AI Volume 1957, January 2001
[12] C. Iglesias, M. Garijo, J. Gonzales, A survey of agent-oriented methodologies, Intelligent Agents IV, Lecture Notes in Computer Science, Springer-Verlag 1555, 1999
[13] N. R. Jennings, M. Wooldridge, Agent–Oriented Software Engineering, in the Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World : Multi-Agent System Engineering (MAAMAW-99), Valencia, Spain, 1999
[14] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos A. Perini, TROPOS: An Agent Oriented Software Development Methodology. Journal of of Autonomous Agents and Multi-Agent Systems. Kluwer Academic Publishers Volume 8, Issue 3, Pages 203 - 236, May 2004
[15] F. Giunchiglia, J. Mylopoulos, A. Perini, The Tropos Software Development Methodology: Processes, Models and Diagrams, Lecture Notes in Computer Science 2585, pp 162-173, Springer 2003
[16] P. Bresciani, P. Giorgini, The Tropos Analysis Process as Graph Transformation System, In Proceedings of the Workshop on Agent-oriented methodologies, at OOPSLA 2002, Seattle, WA, USA, Nov, 2002
[17] H. Mouratidis, A Security Oriented Approach in the Development of Multiagent Systems: Applied to the Management of the Health and Social Care Needs of Older People In England, PhD thesis, University of Sheffield, 2004.
[18] E. Yu, Modelling Strategic Relationships for Process Reengineering, Ph.D. thesis, Department of Computer Science, University of Toronto, Canada, 1995
[19] J. Castro, M. Kolp, J. Mylopoulos, Towards Requirements-Driven Information Systems Engineering: The Tropos project, In Information Systems (27), pp 365-389, Elsevier, Amsterdam - The Netherlands, 2002
[20] A. Fuxman, L. Liu, M. Pistore, M. Roveri, J. Mylopoulos, Specifying and Analyzing Early Requirements: Some Experimental Results. In Proceedings of the 11th IEEE International Requirements Engineering Conference, 8th-12th September 2003, Monterey Bay, California U.S.A

[21] A. Fuxman, Formal Analysis of Early Requirements Specifications, MSc Thesis, University of Toronto, Canada, 2001
[22] P. Bertrand, R. Darimont, E. Delor, P. Massonet, A. Van Lamsweerde. GRAIL/KAOS: an environment for goal driven requirements engineering, In Proceedings of the 20th International Conference on Software Engineering (ICSE'98), IEEE-ACM, Kyoto, April 98
[23] B. Bauer, J. Müller, J. Odell, Agent UML: A Formalism for Specifying Multiagent Interaction, In Agent-Oriented Software Engineering, Paolo Ciancarini and Michael Wooldridge (eds.), Lecture Notes in Computer Science, pp. 91-103, Springer, Berlin, 2001
[24] I. Philp, Can a medical and social assessment be combined? Journal of the Royal Society of Medicine, 90(32), pp 11-13,1997
[25] J. Odell, C. Bock, Suggested UML extensions for agents, Technical report, OMG, December 1999. Submitted to the OMG's Analysis and Design Task Force in response to the Request for Information enti-tled "UML2.0 RFI"
[26] M.D. Beer, R. Hill, W. Huang, A. Sixsmith, An agent-based architecture for managing the provision of care – the INCA (Intelligent Community Alarm) experience, Proceedings of the workshop on Agents Applied in Health Care, at the 15th European Conference on Artificial Intelligence, France-Lyon, 2002
[27] S. M. Lee, A. A. Asllani, S. Trim, An Object-Oriented Approach for Designing Service Scheduling Support Systems, in International Journal on Services and Standards, Vol. 1, No. 1, 2004
[28] M. Krol, D.L. Reich, Object-Oriented Model of a Health Care System, in the Proceedings of the 11th International Symposium on Computer Based Medical Systems (CBMS'98), TX-USA, 1998.
[29] B. Blobel, Application of the component paradigm for analysis and design of advanced health system architectures, in International Journal of Medical Informatics 60(2000), pp. 281-301.
[30] J. Odell, P. Giorgini, J. P. Muller (eds), Proceedings of the Fifth International Workshop on Agent Oriented Software Systems (AOSE'04), N.Y. –USA, July 2004.
[31] Scott A. DeLoach, Modeling Organizational Rules in the Multiagent Systems Engineering Methodology, Proceedings of the 15th Canadian Conference on Artificial Intelligence (AI'2002), Calgary, Alberta, Canada. May 27-29, 2002.
[32] R. Evans, P. Kearney, J. Stark, G. Caire, F. J. Carijo, J. J. Gomez Sanz, J. Pavon, F. Leal, P. Chainho, and P. Massonet. MESSAGE: Methodology for Engineering Systems of Software Agents, AgentLink Publication, September 2001
[33] F. Zambonelli, N. R. Jennings, M. Wooldridge, Organisational Abstractions for the Analysis and Design of Multi-Agent Systems, P. Ciancarini and M. Wooldridge (eds.), Agent-Oriented Software Engineering, Springer-Verlag, Lecture Notes in AI, Vol. 1957, January 2001