Simple Executable Architecture Example

Volleyball Simulation

Mr. M.G. Ball
Joint Staff Operational Research Team

DRDC CORA TN 2005-15
December 2005

Defence R&D Canada
Centre for Operational Research & Analysis

Joint Staff Operational Research Team
Deputy Chief of the Defence Staff

National Defence Défense nationale

Canada

Simple Executable Architecture Example

Volleyball Simulation

M.G. Ball

Joint Staff Operational Research Team

Centre for Operational Research and Analysis

Technical Note

TN 2005-15

2006-01-27

Author

Mr. M.G. Ball

Approved by

Mr. R.W. Funk

Team Leader Joint Staff Operational Research

Approved for release by

Mr. R.G. Dickinson

Director Operational Research (Joint)

The information contained herein has been derived and determined through best practice and adherence to the highest levels of ethical, scientific, and engineering investigative principles. The reported results, their interpretation, and any opinions expressed therein, remain those of the authors and do not represent, or otherwise reflect, any official opinion or position of DND or the Government of Canada.

Abstract

The Joint Staff Operational Research Team (JSORT) is developing a State Machine model using the CORE software package to simulate the concurrent workflows that exist in net-centric operational activities. Modelling architectures in CORE takes time and experience to learn properly so a simple model of a volleyball match was used to practice building a well-known system from start to finish. This report discusses the volleyball model elements, visualization of the match's progress and then assesses how varying player abilities affects the likely outcome of the match. The final chapter discusses possible extensions to the model and details the lessons learned through this process.

Résumé

L'Équipe de recherche opérationnelle interarmées (EROI) élabore actuellement un modèle automate fini à l'aide du progiciel CORE dans le but de simuler des déroulements concurrents de travaux qui surviennent dans des activités opérationnelles réseaucentriques. La connaissance adéquate des architectures de modélisation de CORE exige temps et expérience. C'est pourquoi un modèle simple de partie de volleyball a servi d'exemple lors d'un exercice d'élaboration du début à la fin d'un système connu. Ce rapport présente les éléments du modèle de volleyball, la visualisation de la progression des parties et enfin l'évaluation de l'incidence des habiletés des différents joueurs sur les résultats possibles de la partie. Le dernier chapitre présente les prolongements possibles du modèle et les détails sur les leçons apprises au long du processus.

This page intentionally left blank

Executive Summary

The Joint Staff Operational Research Team (JSORT) is currently in the beginning phase of developing capability engineering simulation tools to support the analysis of executable architectures based on the Department of Defense Architecture Framework (DoDAF) [1]. One of the tools being created is the State Machine model for capabilities such as the Joint Intelligence and Information Fusion Capability (JIIFC) [2]. This modelling is being done with CORE, a system engineering software package developed by Vitech Corporation. After learning the basics of CORE, it was decided that it would be a worthwhile effort to model a simple, well-known system from start to finish as a practice problem.

The example of a volleyball game was chosen because it follows a well-defined structure in terms of which team can do what and when, which results from the fact that there is a limited number of outcomes to any particular action. The aim was to model a volleyball match between two teams, taking into proper account the various possible sequences of activities.

An initial analysis was performed on data from the COREsim transcript to show the progression of resources, namely each team's score and number of games won so far, over time. The sensitivity analysis was done to determine which skills had the greatest impact on the outcome of the match. The analysis was then taken one step farther to see how a wider range of variations in serving ability affected the match.

The report itself describes the details of the volleyball model, beginning with the diagrams which relate the various activities within a volleyball match and moving on to the finer details which make the model execute properly but which are not explicitly shown in the diagrams. These include the resources which the model keeps track of, the logic scripts which were written for certain activities, and the probabilities of different outcomes, typically the success or failure of a given activity. The final chapter discusses the possibility of extending the model to account for external conditions, such as environmental effects, or to be more realistic such as allowing for different outcomes to passes, or different types of hits. It also discusses what was learned through this practice problem.

Ball, M.G. 2005. Simple Executable Architecture Example – Volleyball Simulation. TN2005-15, DRDC CORA.

Sommaire

L'Équipe de recherche opérationnelle interarmées (EROI) est actuellement à l'étape de départ dans l'élaboration d'outils de simulation d'ingénierie de la capacité qui servent à appuyer l'analyse des architectures exécutables fondées sur le cadre d'architecture du ministère de la Défense (CAMDN) [1]. Un des outils en cours de développement est le modèle automate fini pour les capacités telles que la capacité de fusion de l'information et du renseignement interarmées (CFIRI) [2]. Cette modélisation se fait à l'aide de CORE, un progiciel d'ingénierie système élaboré par Vitech Corporation. Après avoir étudié les principes fondamentaux de CORE, il a été décidé qu'il serait utile de faire le modèle complet d'un système simple et connu à titre d'exercice.

Le modèle d'une partie de volleyball a été choisi en raison de sa structure définie dans laquelle il est possible de définir ce que chaque équipe peut faire et quand elle peut le faire. Cette modélisation est concevable parce qu'il n'y a qu'un certain nombre de résultats possibles à une action. L'objectif était de faire un modèle de partie de volleyball entre deux équipes, sans oublier les diverses séquences d'activités envisageables.

Une première analyse a été effectuée sur les données provenant de la transcription du COREsim pour démontrer la progression des ressources, notamment la marque de chaque équipe et le nombre de parties remportées jusqu'à maintenant. L'analyse de sensibilité a été effectuée afin de déterminer quelles compétences avaient la plus grande incidence sur les résultats de la partie. Une autre analyse est allée plus loin et a tenté de déterminer comment une large gamme de variations dans les capacités du service affectaient le résultat des parties.

Le rapport à lui seul décrit en détail le modèle du volleyball, en commençant par les diagrammes des diverses activités dans une partie de volleyball pour ensuite présenter les détails qui permettent au modèle de fonctionner adéquatement, mais qui ne sont pas démontrés de manière explicite dans les diagrammes. Parmi ces détails se trouvent les ressources pour lesquelles le modèle assure le suivi, les scripts logiques qui ont été rédigés pour certaines activités, ainsi que les probabilités de divers résultats, généralement la réussite ou l'échec d'une activité donnée. Le dernier chapitre présente la possibilité de prolongement du modèle pour tenir compte des conditions externes, comme l'environnement, ou pour être plus réaliste en prévoyant différents résultats aux passes ou les différents types de coups frappés. Il présente également les leçons apprises tout au long de l'exercice.

Ball, M.G. 2005. Simple Executable Architecture Example – Volleyball Simulation. TN2005-15, DRDC CORA.
.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

The Joint Staff Operational Research Team (JSORT) is currently in the beginning phase of developing capability engineering simulation tools to support the analysis of executable architectures based on the Department of Defense Architecture Framework (DoDAF) [1]. One of the tools being created is the State Machine model for capabilities such as the Joint Intelligence and Information Fusion Capability (JIIFC) [2]. This modelling is being done with CORE, a system engineering software package developed by Vitech Corporation. After some amount of learning about what can be done within CORE, it was decided that it would be a worthwhile effort to model a simple, well-known system from start to finish as a practice problem before diving into the State Machine in an attempt to learn the software and the model at the same time.

The example of a volleyball game was chosen because it follows a well-defined structure in terms of which team can do what and when, which results from the fact that there is a limited number of outcomes to any particular action. This also happens to be a system which the author is particularly familiar with, making knowledge of the software the limiting factor in modelling the architecture.

This practice model also served as familiarization process prior to taking a CORE Training Course used to complete the learning process. The work described here is used the information from the COREsim 4.0 Class Materials [3] and a Collaborative Capability, Definition, Engineering and Management (CapDEM) CORE Webinar demonstration on November 3, 2005. The CORE course was taught in CORE 5.1 [4] and volleyball model runs in CORE 5.1.

1.2 CORE Terminology

CORE has many different diagrams available to illustrate the sequence of activities[1]. The one used by most modellers is the Enhanced Function Flow Block Diagram (EFFBD). The difference between an EFFBD and a normal Function Flow Block Diagram is that the EFFBD will also illustrate data flow between different points on the diagram by use of information flows or triggers. In the case of Operational Activities, this data is referred to as Operational Information (OI). In many cases, and all cases in this model, the activity treats the OI as a "trigger" which it waits for before executing. An example from this model is that the defending team must wait for the information that the attacking team is performing a hit before they can decide whether or not to attempt a block. In CORE terms, the activity of hitting produces the trigger that the activity of blocking must wait for that before it can execute.

[1] The types of diagrams available for an activity are: Element Relationships, Hierarchy, Function Flow Block Diagram, Enhanced Function Flow Block Diagram, N2, IDEF0, and IDEF0-A0.

The CORE class used most often in this simulation is the Operational Activity (hereafter referred to as activity), which is used to represent a person or group (in this case usually a volleyball team) doing something. Activities are represented in the EFFBDs as rectangular boxes with a number and a label.

If an activity, or series of activities, is to be repeated a certain number of times, it can be placed within an iterate structure in the CORE diagrams. This is represented by the letters IT placed before and after the activities which are to be repeated, with an arrow pointing back from the second to the first IT symbol. Similarly, if activities are to be repeated until a breaking condition is met, they can be placed within a loop, which looks much like an iterate structure but is represented by LP instead of IT. The loop repeats until a loop exit, represented by LE is encountered.

1.3 Aim

The goal of this exercise was to practice using the functionality of CORE which was recently learned through the practice problems. The specific aim was to model a volleyball match between two teams, taking into proper account the various possible sequences of activities. As an extension to this, some analysis was undertaken to pull information out of the simulation. Specifically, the effects of varying the probability of success of various skills were studied.

1.4 Scope

The next chapter of this report describes the details of the volleyball model, beginning with the diagrams which relate the various activities within a volleyball match and moving on to the finer details which make the model execute properly but which are not explicitly shown in the diagrams. These include the resources which the model keeps track of, the logic scripts which were written for certain activities, and the probabilities of different outcomes, typically the success or failure of a given activity.

After this some simple analysis of the results of the simulation is presented. This demonstrates how COREsim can be used to study how a model's outcome depends on various aspects of the model.

Finally, the report discusses the possibility of extending the model to be more realistic, or to account for external effects, as well as how this exercise was useful in developing expertise in the use of CORE.

2. The Volleyball Model

2.1 Block Diagrams

At the highest level, a volleyball match is summed up as several games being played repeatedly, with each team's score being reset to zero before each game, and the first serve of each game being given to whichever team did not serve first in the previous game. This behaviour is modelled in the EFFBD in Figure 1. In this case, a step is added after each game where the match score (the number of games won by each team so far during the match) is being checked.

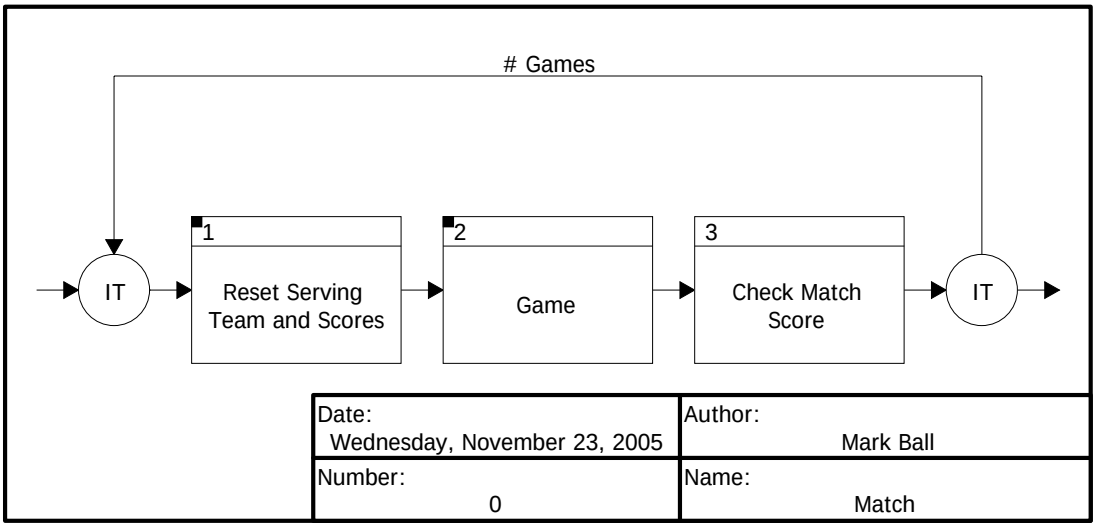

Figure 1. Match Activity Diagram

The next two sections describe the lower-level behaviour of the "Reset Serving Team and Scores" and "Game" activity (there is no lower level behaviour to checking the match score).

2.1.1 Reset Serving Team and Scores

Each team's score is held as a resource and so the first step in the "Reset Serving Team and Scores" activity (illustrated in Figure 2) is a sub-activity, "Reset Scores", which captures both of these resources. The amount captured is exactly the same as the total amount available, leaving both resources at zero.

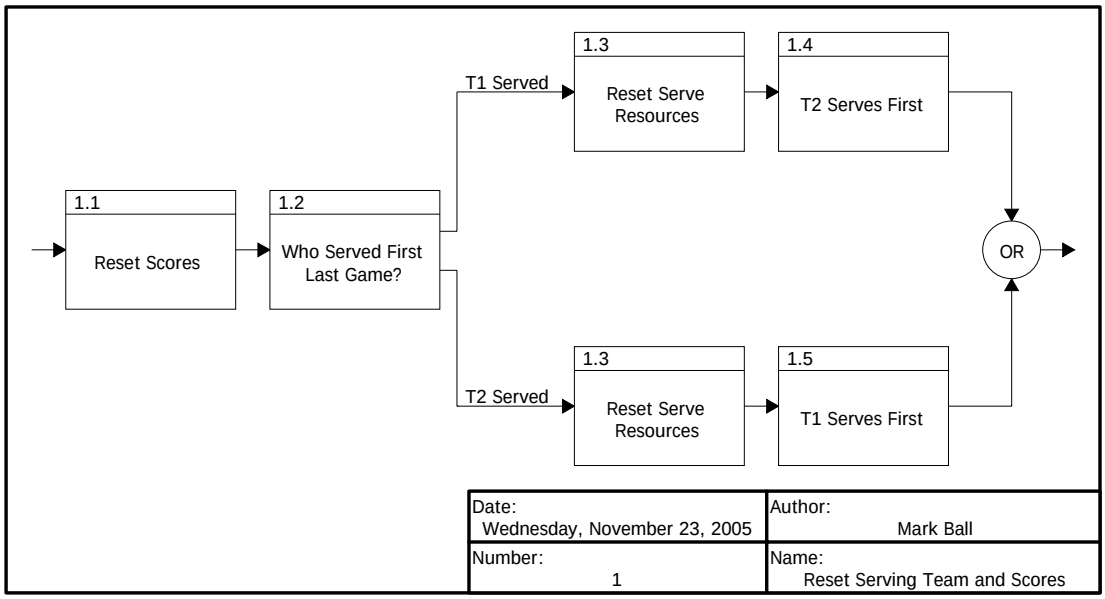

Figure 2. Reset Serving Team and Scores Activity Diagram

The next step, that of determining which team gets to serve first, is a little more complicated. First, the team to serve first is determined by looking at the previous game. Before the first game of the match, it will be determined that Team 2 served first during the previous game and so it is now Team 1's turn. Once it is determined which team served first, either the "T1 Served" or "T2 Served" branches shown in Figure 2 will be followed. After this is done, it is safe to 'forget' about the previous game and who served first during the most recent rally. Both of these values are held as resources which are consumed by the "Reset Serve Resources" activity. Finally, if the "T1 Served" branch was taken, the "T2 Serves First" activity will execute, and vice-versa. "T2 Serves First" will set both resources which were just consumed by "Reset Serve Resources" to 2, or "T1 Serves First" will set them to 1. The "Serving Team" resource can now be used to determine which team will serve next and the "First Serve" resource will be checked the next time "Who Served First Last Game?" runs.

2.1.2 Game

As with the match, a single volleyball game can be summed up very simply at a high level. In this case, a game consists of several rallies repeated until conditions necessary for the end of the game are met (as illustrated in the EFFBD of Figure 3). These conditions are checked at the end of each rally with the possible outcomes of this check being that either Team 1 or Team 2 has won, or another rally must be played.

The majority of the simulation takes places within the "Rally" activity, which, as the name implies, represents a rally between two teams which ends when one of the teams scores a point. This activity, illustrated in Figure 4, begins with a sub-activity which models the serve (Figure 5) and which is followed by possessions (Figure 6) and possibly blocks (Figure 7). The rally

will end when one of the last two branches in Figure 4, representing either team scoring, is triggered.

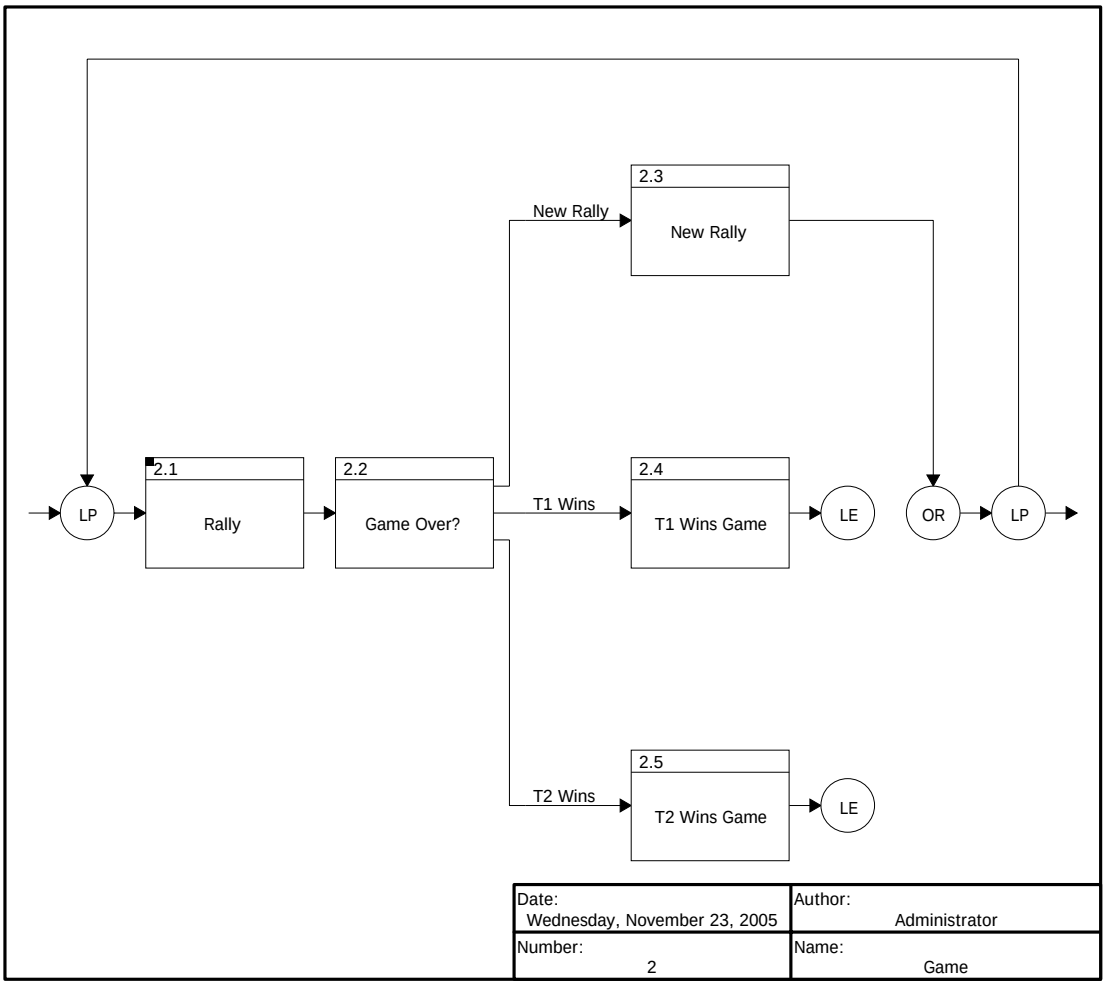

Figure 3. Game Activity Diagram

The "Serve" activity represented in Figure 5 begins by deciding which team will serve based on the "Serving Team" resource. This resource is set before the game as described in Section 2.1.1 and is consumed by the functions representing teams serving, and produced by the functions representing teams scoring a point. Values of 1 are consumed / produced by Team 1 and values of 2 are consumed / produced by Team 2. Once the serving team is chosen, they may hit the ball in, triggering the other team's possession, or out, triggering a point scored by the other team.

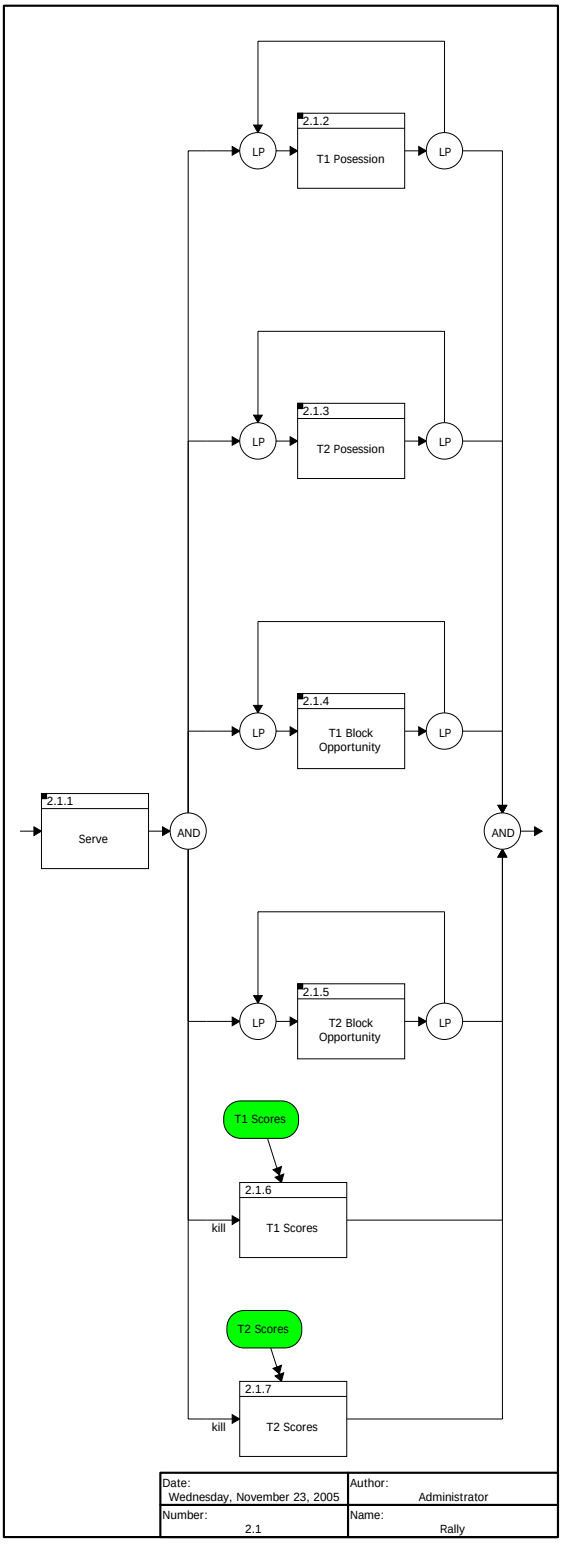

Figure 4. Rally Activity Diagram

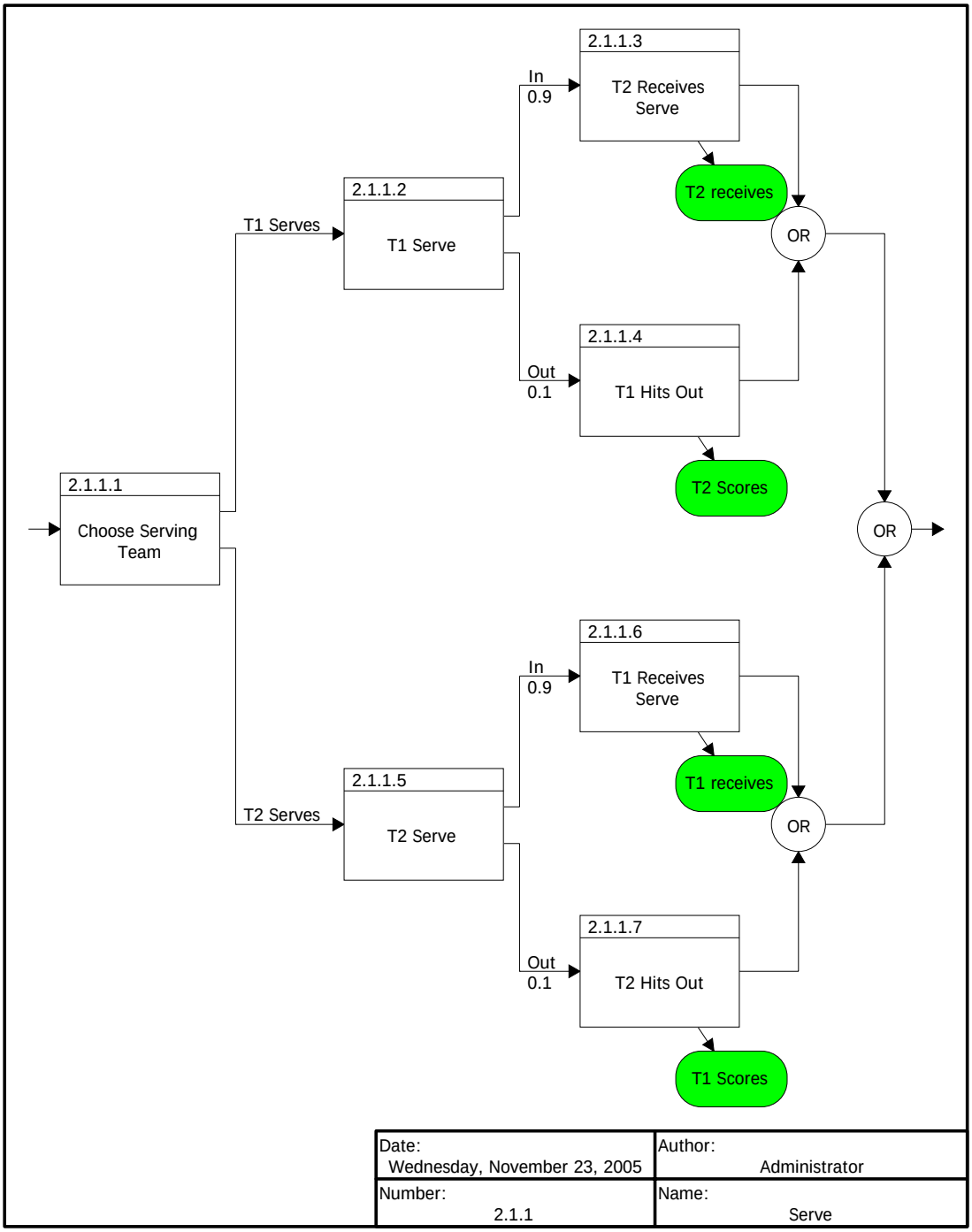

Figure 5. Serve Activity Diagram

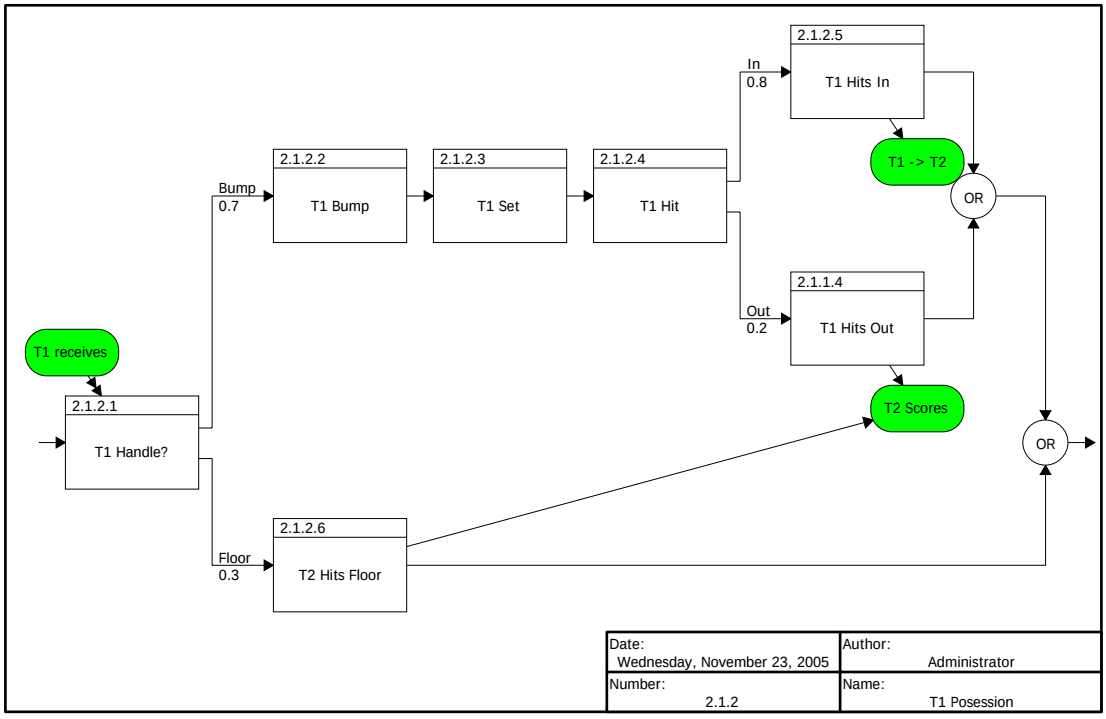

Figure 6. Possession Activity Diagram

When the outcome of the serve is determined, the simulation enters a six branch AND structure, the four first branches of which are contained within loop structures. The final two branches represent either team scoring a point and can be triggered either by that team hitting the ball onto the floor in their opponents court, or by their opponent hitting the ball out of play. Both of these activity are on branches with the kill status turned on, which causes the AND structure (and therefore the rally) to end.

The other 4 branches of the AND structure represent the team's actions which may be repeated throughout the rally. There is one branch representing each team's possession of the ball, where they attempt to bump, set, and hit the ball, and another branch per team to model their attempts to block their opponents' hit. The possession and block activity are illustrated in Figure 6 and Figure 7, respectively. Note that these figures apply to Team 1's possession and block attempt, though the activity for Team 2 are identical, but with the teams reversed on each sub-activity.

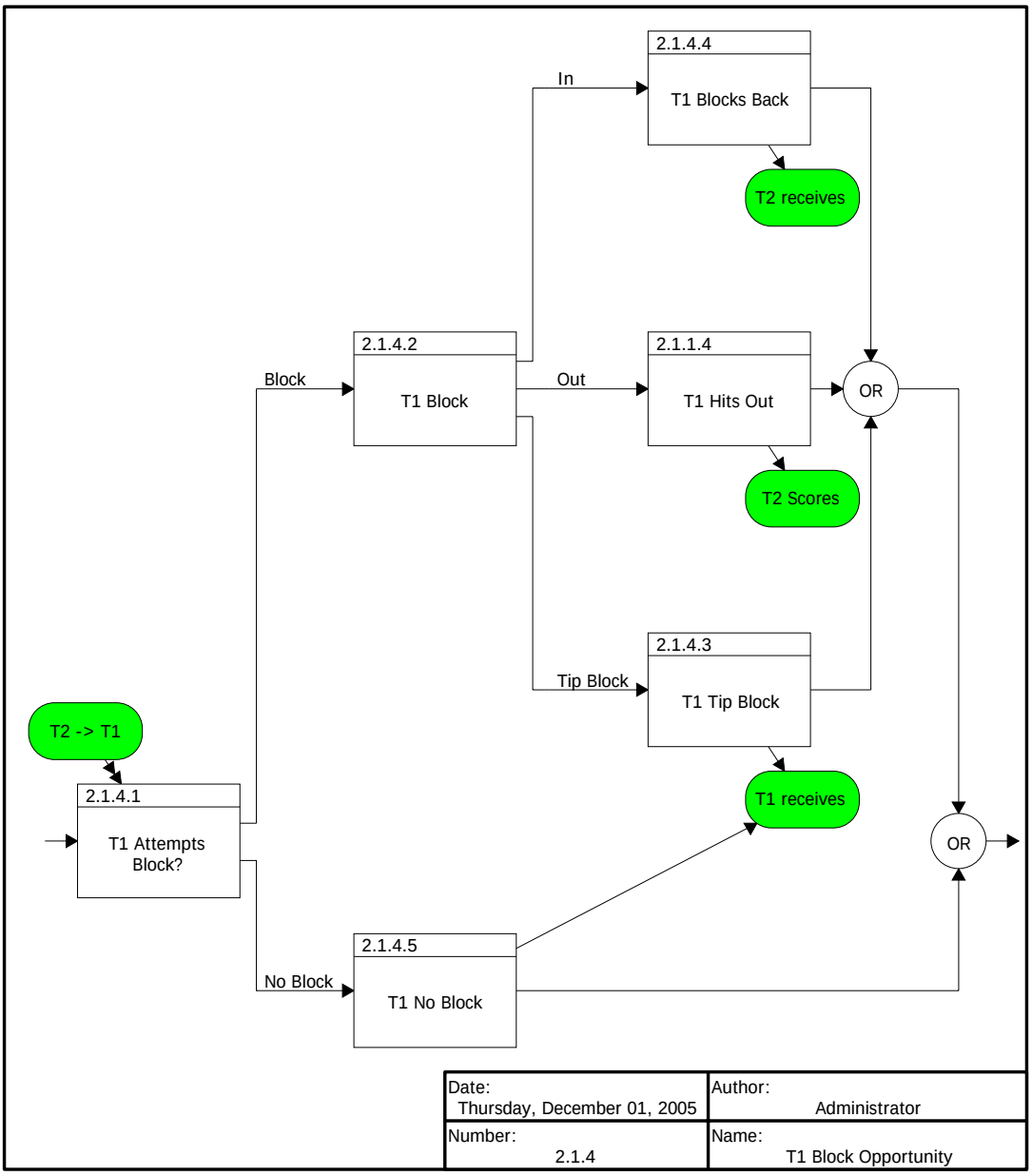

Figure 7. Block Activity Diagram

2.2 Triggers

In the "Rally" activity, it can be seen that the sub-activity representing each team scoring a point are triggered by Operational Information (OI) of the same names. These are produced by the appropriate activities, as noted in the text accompanying Figure 4. The "T1 receives" OI triggers Team 1's possession of the ball, and is produced by Team 1 receiving the serve, touching the ball on a block attempt, not blocking a hit by Team 2, or by Team 2 blocking a hit back to Team 1. The "T2 -> T1" OI triggers Team 1's decision whether or not to attempt a block, and is produced by a successful hit by Team 2. These two OIs also exist in their reverse forms where

Teams 1 and 2 are switched. Both versions of these two OIs can be seen at several points in Figure 5 through Figure 7.

2.3 Resources

2.3.1 Serving Team

As stated earlier, the "Serving Team" resource is used to determine which team will serve at the beginning of each rally. It is produced and consumed, in the appropriate numbers, by either team scoring and serving, respectively. It is also consumed and reproduced as appropriate before each game so that whichever team received the first serve of the previous game will serve at the beginning of the upcoming one.

2.3.2 First Serve

The "First Serve" resource keeps track of which team served first at the beginning of the previous game. If used during a game it identifies which team served first to begin the current game. It is consumed before each game by the "Reset Serve Resources" activity and then produced by either "T1 Serves First" or "T2 Serves First", just as the "Serving Team" resource is. The difference between these two resources is that "First Serve" does not change during a game.

2.3.3 Scores

The "T1 scores" and "T2 scores" activities also produce resources ("T1 score" and "T2 score") representing each team's score. These resources are used in determining when a game is ended, and can be looked to at any point in the simulation to determine the current score. Before each game they are consumed by the "Reset Scores" activity.

2.3.4 Ball

The "Ball" resource represents the volleyball and is captured by either team serving, passing, hitting or blocking. As there is only one ball in play, only one of these activities can happen at once (note that due to the sequential nature of the events being simulated, no two of them ever happen at once anyway).

2.3.5 Team Games

The "T1 Games" and "T2 Games" resources are the number of games won by each team. They are produced by the "T1 Wins Game" and "T2 Wins Game" activity which are located on the appropriate exit branches of the "Game Over?" activity. These can be used to determine the winner of a match and

are also useful in statistical studies involving many games. Particularly, they are captured by the “Check Match Score” activity after each game.

2.4 Logic Scripts

2.4.1 Exit Logic

Three exit logic scripts are used in this simulation. The first is part of the “Who Served First Last Game?” activity which checks the value of the “First Serve” resource. If that value is 1, then Team 1 served first to start the last game. For any other value, Team 2 is determined to have served first during the previous game. The reason for this is so that before the first game, the value of “First Serve” will be zero which will lead to Team 1 having the first serve of the match. The second takes place in the “Choose Serving Team” activity which uses the “Serving Team” resource and similar logic to decide which team will serve.

The third exit logic script takes place in the “Game Over?” activity and it begins by asking if Team 1 has at least 25 points, and if so do they also have more than 1 more point than Team 2? If the answer to both of these questions is yes, then the game is over and Team 1 wins, otherwise the questions are asked again about Team 2. Again if the answers are both yes, the game is over but this time Team 2 wins, otherwise a new rally must be started.

2.4.2 Resource Logic

Several activities must consume or capture resources in their entirety, and to do so they must first check to see how much of the resource in question is available. In these cases scripts were written for the consumes / consumed by or captures / captured by relationships such that the activity in question first checks the value, then consumes that amount, of the resource.

The “Reset Serve Resources” activity consumes the “First Serve” and “Serving Team” resources according to the above procedure. The “Reset Scores” activity does the same to the “T1 Score” and “T2 Score” resources.

The “Check Match Score” activity captures the “T1 Games” and “T2 Games” resources, and the “Game Over?” activity captures the “T1 Score” and “T2 Score” resources in their entirety. The reason this is done is so when the simulation is run, the transcript will record the values captured. This way, a quick look at the transcript will reveal the game score or match score at any given time.

2.5 Probabilities

All probabilities are arbitrarily assigned to represent teams that are more likely than not to accomplish any attempt to do something. In most cases there are two possibilities: a serve can be in or out, a team gaining possession can handle (bump)

the ball or miss it (it hits the floor), a hit can either go in or out. There are three possible outcomes to a block: it can go in on either team's side of the net (a tip block goes to the blocking team, blocking the ball in sends it back to the attacking team) or it can go out. Also, a team may not block every chance it gets. This possibility combines both a block attempt where the blocker did not touch the ball, and the defending team simply deciding not to attempt a block at all.

Unsuccessful passes have not been modelled. Instead, they are considered to fall under missing the ball on attempting a bump pass or hitting the ball out on the attack hit. As well, the possibility of sending the ball across the net on the first or second hit is not considered. These cases represent outcomes with zero probability for the purposes of the simulation. They would be simple in practice to add to the model: Both the "Bump" and "Set" activity could have three possible exits: pass, send the ball across the net, or send the ball out of play. The increased modelling fidelity would incur a corresponding increase in model complexity and execution time.

2.6 CORE Simulator

The CORE simulator allows the model to be executed and the results of this execution are viewed graphically, with time on the horizontal axis. Which elements of the simulation are displayed can be chosen by the user, and a sample of the results is illustrated in Figure 8. Grey blocks represent the progression of resource levels over time, yellow represents an activity waiting for a trigger, green represents an activity being executed, and white with green hatching represents an activity which is being executed through lower level activities.

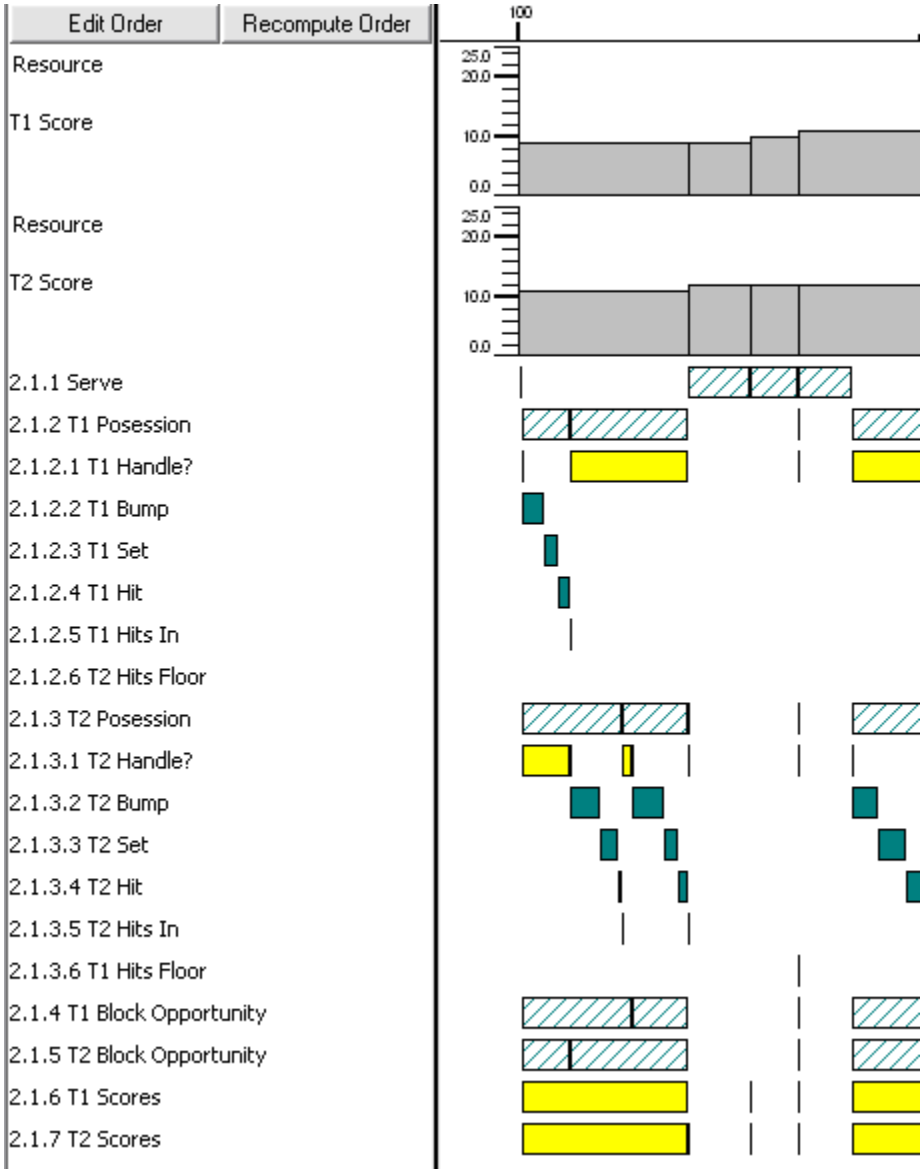

Figure 8. Simulator Results for Four Serves

3. Analysis

3.1 Behaviour Over Time

COREsim has the ability to produce a transcript of events which can be exported as a comma separated values (.csv) file. The catch is that the transcript only records the values that were added to or removed from resources, not what the resulting resource level is. As mentioned in Section 2.4.2, this is the reason for resource checking activities which capture resources in their entirety. Once the transcript is created and saved as a .csv file, it can be imported into a spreadsheet program such as Excel. It helps to sort the data to isolate those lines of the transcript which describe the resource levels at given times.

The first step is to sort according to the column which contains the name of the activities consuming, capturing, or producing resources. This way the resource checking activities can be isolated from other activities, including activities which manipulate resources without checking their totals.

The next column to sort by is the one containing the resource name, as we are looking to monitor the progression of one resource at a time. Note that if all resource checking activities only check one resource, then this step produces no refinement over the previous one. This could easily be done for the “Check Match Score” activity, for example, by breaking it down into two activities, “Check T1 Games” and “Check T2 Games”. Any activity which captures resources will also release them when it completes, producing two lines in the transcript where only one is needed.

The next step then is to sort by the column which differentiates between capturing and releasing resources. Now all the data should be grouped into categories that are to be monitored separately. The final step is to sort the data within those categories by time. Plotting the values captured within categories, as function of time, will produce results such as those in Figure 9.

Figure 9 shows the results of the above described method applied to an 8 game run of the volleyball simulation. Here the green lines show the progression of the match until Team 2 won 5 games to Team 1’s 3. The red and blue lines show the progression of actual points on the scoreboard throughout the match.

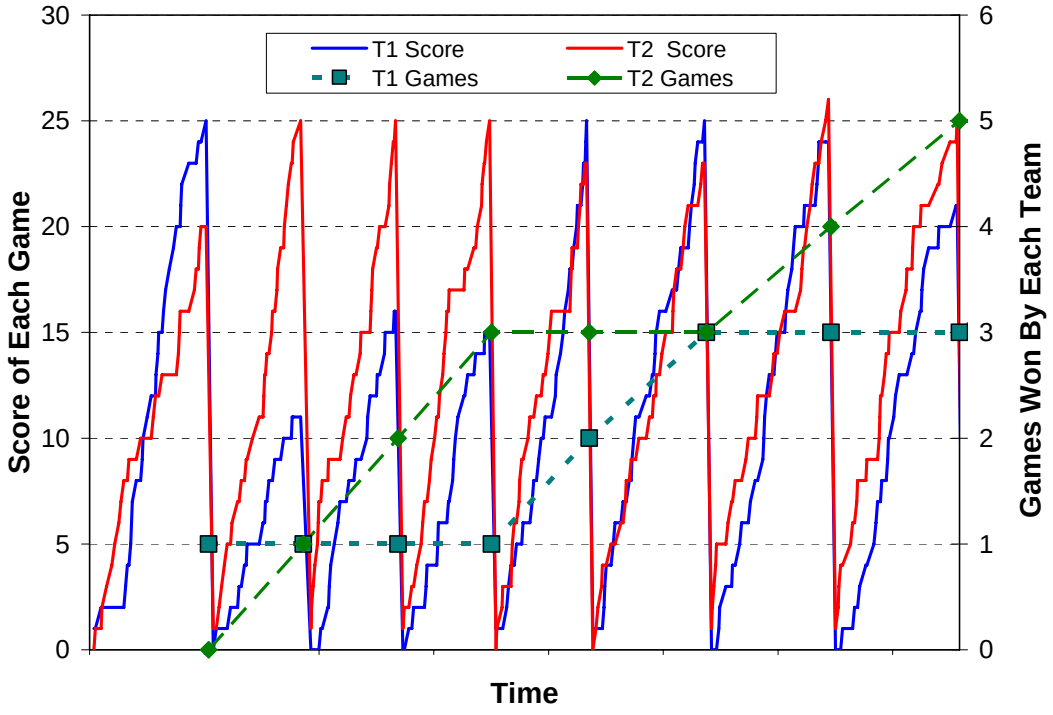

Figure 9. Simulation Results Over Time for Volleyball Match (First to Win 5 Games)

3.2 Sensitivity Analysis

All of the probabilities assigned to the various exit branches were set to equal probabilities and the simulation was run for a 50 game match. This was done several times using different seeds to the random number streams until a seed was found which produced nearly even results for the two teams. Specifically, Team 1 won 26 games to Team 2's 24. These same seeds were then used for the rest of the sensitivity analysis, meaning the results presented here are based on the same streams of random numbers being used every time.

3.2.1 Sensitivity to Particular Skills

The first sensitivity analysis was to determine which skills had the greatest impact on the outcome of the match. For each skill, all of the probabilities not related to that skill were set equal, then Team 1 was given a 60% chance of success at that skill and Team 2 was given a 40% chance. After the simulation was run, these two probabilities were reversed and the simulation was run again. When it came to the three possible outcomes of a block, one exit was given a 40% chance and the other two each had a 30% chance. As this did not produce significant change in the match results, these probabilities were changed to a 60% chance on one exit and a 20% chance on each of the other two. Specific details relating to skills with two possible exits are given in Table 1 and illustrated in Figure 10. Details relating to blocking, with three possible exits, are shown in Table 2.

Table 1. Match Outcomes in Response to Single Skills With Two Exits

Improved Skill	Team 1 Wins	Team 2 Wins
T1 Serve	43	7
T2 Serve	4	46
T1 Handle	36	14
T2 Handle	24	26
T1 Hit	31	19
T2 Hit	18	32
T1 Attempt Block	27	23
T2 Attempt Block	19	31

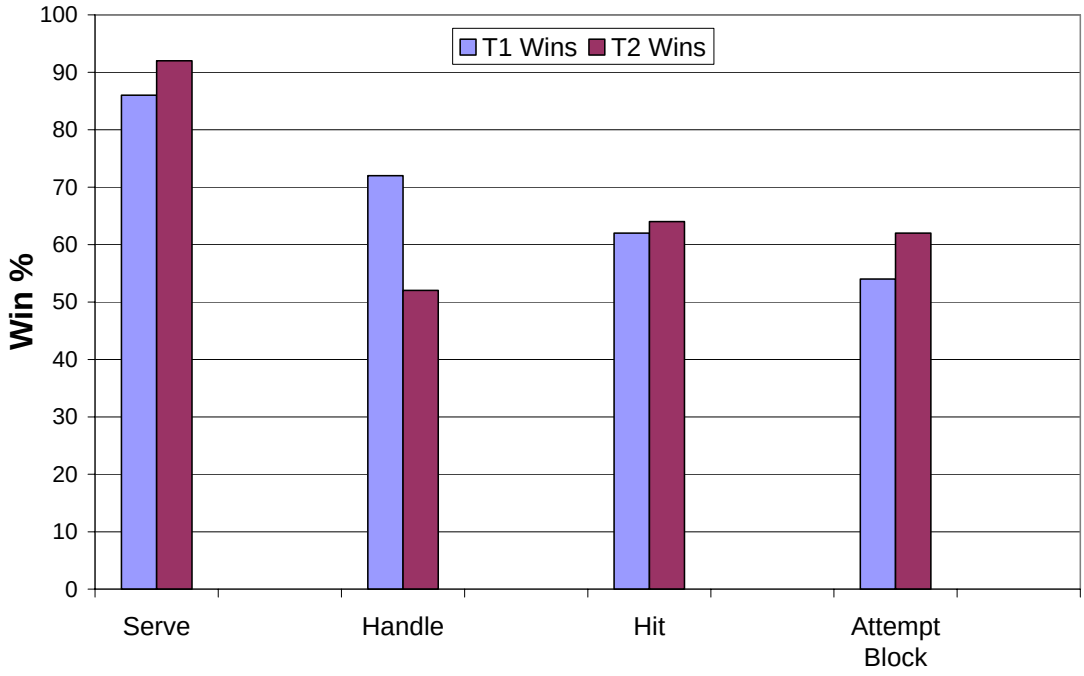

Figure 10. Match Outcomes in Response to Single Skills With Two Exits

Table 2. Match Outcomes in Response to Blocking Skill

Most Likely T1 Exit	Most Likely T2 Exit	T1 Wins	T2 Wins
In (.4)	Out (.4)	21	29
Out (.4)	In (.4)	24	26
In (.4)	Tip (.4)	22	28
Tip (.4)	In (.4)	27	23
Tip (.4)	Out (.4)	24	26
Out (.4)	Tip (.4)	26	24
In (.6)	Out (.6)	28	22
Out (.6)	In (.6)	21	29
In (.6)	Tip (.6)	21	29
Tip (.6)	In (.6)	28	22
Tip (.6)	Out (.6)	29	21
Out (.6)	Tip (.6)	19	31

3.2.2 Special Case of Blocking

The result where one team has a 60% chance of blocking in and the other has a 60% chance of a tip block seems counterintuitive. For instance, when Team 1 tends to block in and Team 2 tends to tip block, both of these tendencies lead to Team 2 having possession. At this point, there is a 50% chance that they will mishandle the ball, and even if they make the pass, there is a 50% chance they will hit it out. This means that no matter which team is blocking, the most likely outcome is a possession by Team 2, which has a 75% chance of Team 1 scoring within the next two random outcomes. However, this seems to lead to Team 1's record dropping from 26-24 to 21-29. Similarly, in the reverse situation where blocking seems to favour Team 2, Team 1's record improves to 28-22.

A closer look at the logic reveals that Team 1 blocks the ball back to Team 2 a total of 22 times, they block it out 23 times, and they tip block 50 times during the 50 game match. The numbers for Team 2 are 62, 23, and 25 respectively. In other words, the most likely result of each team's block occurs less than twice per game. This is not entirely surprising as several things must happen in a rally before a block is triggered. Specifically, the serve must be in, the handle must result in a bump, the hit must be in, and the block must be attempted. These are each 50% chances, leading to less than a 7% chance of a block during any given rally.

It seems then that the effect that good blocking has on a match is actually a side-effect of changing how the random number stream is applied to the skills following the block. In order to see more blocks per game, one could increase the chance of success of all the other skills. For the purpose of this study, it is sufficient to say that the outcome of blocking has less effect on the results of the match than do the outcomes of the other skills which are applied more often.

3.2.3 Extended Serve Sensitivity

In the previous analysis the serving ability had the largest impact on the outcome of the match, so this portion of the analysis was taken a step farther to see how variations in serving ability affected the match. As in the previous section, all probabilities were set equal except for those relating to Team 1's serve. The probability of Team 1's serve going in was then varied over several runs of the simulation, producing the results of Figure 11.

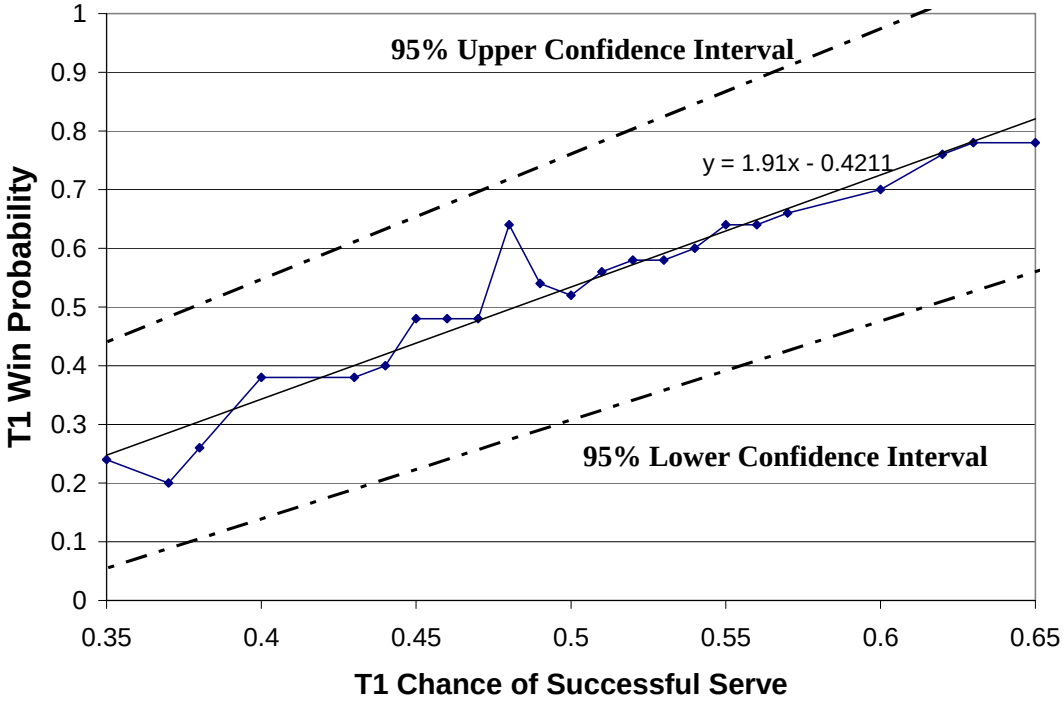

Figure 11. Match Sensitivity to T1 Serve Ability

The curve of Figure 11 tends to be smoother above a 50% chance of a successful serve. This implies that the more reliable the serve, the more predictable its effect on the final outcome. Likewise, the increased variability in the lower left quadrant illustrates how the outcome depends increasingly on the string of random numbers rather than skill. Note that even though the curve appears to be linear it is obviously a local effect in the region around 35-65 % chance of successful serves. More data would have to be taken to examine the behaviour of the model outside of this range.

4. Way Ahead

4.1 Possible Extensions

4.1.1 Environment

It has been suggested that adding biases to the teams' abilities to account for the environmental effects of outdoor games, such as wind or intense sunlight, could be done. These factors are difficult, though certainly not impossible, to quantify, particularly because the weather and players' reactions to it are not well known. However, skills have been randomly assigned to the teams in this simulation and the same could be done for environmental effects. Some creativity would have to go into accounting 180 degree shift in wind and sunlight direction when teams switch sides between games. A simple solution would be to have two games within the iteration loop in the "Match" activity diagram of Figure 1, the first would have probabilities based on Team 1 facing a certain direction, and the second based on Team 2 facing that direction.

4.1.2 Pass Outcomes and Hit Types

As mentioned in Section 2.5, it is assumed that bumps lead to sets, which lead to hits. Bumps and sets could be given three exit branches: pass, send the ball over the net, send the ball out of play. These are not considered in the current model as the second option leads to the same end result as the first, and the third option leads to the same end result as missing the bump completely (which is accounted for in the "Handle?" activity). Though not providing new outcomes, these exit branches would provide information as to how the outcomes are achieved. If it was felt that this extra information was worth the extra complexity, it would be simple to add it to the model.

It would also be possible to model different ways of hitting the ball over the net. Serves, spikes, tips, and free balls are not equally easy for the defending team to handle. The last three of these could be included as possible outcomes of each team's possession and each of the four would trigger different versions of the "Handle" activity, each version having a different probability of success. Alternatively, they could each lead to the same "Handle" activity, which would have an exit logic script to determine the odds of success based on the type of hit. This comes down to a choice between more complicated CORE diagrams, or more complicated Smalltalk coding.

4.1.3 Further Analysis

The analysis of Chapter 3 could be extended several ways. For example, instead of making the default case one where all exits from any given activity are equally likely, all outcomes could be given set probabilities, say 90%

success and 10% failure, to see how varying one skill affects the outcome of a match between teams at higher skill levels. Note that success of any given activity keeps the rally going, whereas failure, with the exception of attempting a block, results in a point for the other team. Because of this, higher skilled teams are expected to have longer rallies.

4.2 Lessons Learned

The goal of this exercise was to practice using CORE to model a well-known system and so the most important result is the experience gained with using the COREsim software application. Perhaps the most obvious benefit of doing the example was learning how to quickly create relationships between various classes, and between different elements within a class. A quick look at CORE's Project Explorer window shows that the simulation includes 50 Operational Activities, 14 Exits, 7 Resources, and 6 Operational Information nodes. Assigning relationships between classes became somewhat instinctive with practice.

Organizing activities to decompose them into different levels of detail also has benefits outside of simple cosmetics. Viewing a small group of activities at a time is not only easier on the eyes, but when done properly it also helps to make the logic more obvious. The lesson here is that it is good practice to do the initial modelling at a high level first and then move to the next level of detail. Specific sub-activities of one high-level activity could also be modelled first, and then one could move on to relating that high-level activity to others within the system. In summary, working either top-down or bottom-up are both acceptable and are better than trying to do all levels of detail at once.

SMALLTALK, the language used to produce scripts within CORE, is a new programming language to the author. Though the exit logic scripts are based on those from other examples, it was encouraging to see that modifications specific to this model could be made to produce expected results. It also transpired that resource consumption scripts are not too far a stretch from exit logic scripts and it was beneficial to learn the differences within a simple model.

There are many ways of customizing the way CORE displays information in the diagrams and the simulator. Generally, removing unwanted information will make it easier to focus on key attributes as well as speed up the simulator. Part of the learning process was to discover what is worth adding to or removing from the display, when is the best time to do so, and how to do so relatively quickly.

The volleyball model was a simple but effective way to learn about CORE software application while dealing with issues associated with building and analyzing executable architecture capabilities. Other potential users of CORE would benefit from building their initial example model based on an architecture with which they are most familiar.

4.3 Interest to Operational Research

One of the big selling points for the CORE software package is the ability to run an executable model in COREsim. In this example, the simulator has been used to generate probabilistic results of the progression of a volleyball game. This same approach can be used to study the behaviour of Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) systems such as the State Machine model of JIIFC mentioned in the introduction. The ability to study the progression of results over time, as well as seeing the effect of changing parameters within the system, is key to understanding the system itself. In this study, this was done to examine the changing scores as the match progressed, and the dependence that the model had on various skills. In the case of JIIFC for example, this same technique can be used to study the efficiency of interactions between operators and how that depends on operating procedures.

4.3.1 Why CORE?

Aside from the quantitative analysis that can be facilitated by CORE, there are a few other aspects that make it a software package worth being familiar with. At the most basic level, CORE lets the user actually see the interactions that are necessary to make the system work. In the volleyball model this capability was used to describe and visualize the dependence of various activities on the outcomes of previous ones, as illustrated by the activity diagrams presented previously. This application will carry over to any system where several interdependent activities are occurring.

The fact that CORE can be used to demonstrate that a modelled architecture is executable. This is a great step toward ensuring that the system being modelled is feasible, or that the model for a system already in existence is valid. There are several elements in the volleyball simulation that, if not properly accounted for, would cause execution problems in COREsim. For example, if hitting the ball out of play did not trigger a point being scored, the simulator would encounter an error at that point. Similarly, if a model were created to represent the exchange of information between military operators, any key step missing in the model would likely be pointed out in the simulation.

A final selling point for CORE is its ability to produce DoDAF views which are used to facilitate communication of the architecture between the parties which designed it, and those which will be implementing and using it. These views are produced through scripts that form part of the software package and result in documentation that follows a standard format. Aside from producing the documentation necessary to take the architecture from the design to the implementation phase, these scripts help ensure that the information contained in the CORE model is complete. If the information required to produce a particular DoDAF view is missing, the document produced by the script will be blank.

5. References

1. Stenbit, J.P. *DoD Architecture Framework (DoDAF) Version 1.0* (http://www.defenselink.mil/nii/doc/DoDAF_v1_Memo.pdf). February 9, 2004. Composed of two volumes and a deskbook linked below.
    - DoDAF Version 1.0 - Volume I (www.defenselink.mil/nii/doc/DoDAF_v1_Volume_I.pdf)
    - DoDAF Version 1.0 - Volume II (www.defenselink.mil/nii/doc/DoDAF_v1_Volume_II.pdf)
    - DoDAF Version 1.0 - Deskbook (www.defenselink.mil/nii/doc/DoDAF_v1_Deskbook.pdf)

2. FUNK, R.W. and SORENSEN, R.(Vitech), State-Machine Modeling of TPED and TPPU, Presented to National Defense Industry Association 8th Systems Engineering Conference, SanDiego, 27 October 2005, 65 slides.

3. Vitech Corporation. *COREsim® 4.0 Class Materials – Modelling and Simulation with CORE*. Vienna, Virginia. May, 2002

4. Vitech Corporation. *Model-Based System Engineering with CORE® – Course Material*. Vienna, Virginia. December, 2004

6. Glossary

Technical term	Explanation of term
C4ISR	Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance
CapDEM	Collaborative Capability, Definition, Engineering and Management
csv	Comma separated values (input and output file option from Excel)
DoDAF	Department of Defense Architecture Framework
EFFBD	Enhanced Function Flow Block Diagram (with Triggers)
FFBD	Function Flow Block Diagram
IT	Iterate
JIIFC	Joint Intelligence and Information Fusion Capability
JSORT	Joint Staff Operational Research Team
LE	Loop Exit
LP	Loop
OA	Operational Activity
OI	Operational Information

This page intentionally left blank

7. Distribution list

3552-1 (DRDC CORA/DOR(Joint))

22 December 2005

Distribution List

DISTRIBUTION OF CORA TECHNICAL NOTE TN 2005-15

Reference: A. CORA Technical Note TN 2005-15, "Simple Executable Architecture Example – Volleyball Simulation", M.G. Ball, December 2005 (enclosed)

1. The attached reference provides a concise example of a simple executable architecture of a volleyball match using the CORE software application. It was done as a learning experience but it also offers a concise learning guide for anybody needing a simple introduction to the modeling issues encountered with CORE.

2. The area where this paper goes beyond the norm is that it includes an analysis segment that illustrates what can be done to exploit the CORE simulator results. The visualization and sensitivity analysis techniques shown here will be expanded upon in future models.

3. As mentioned in the paper, the Joint Staff Operational Research Team (JSORT) will be leveraging the CORE knowledge demonstrated here to help develop simulation tools to model C4ISR architectures.

4. Questions and/or comments on the technical note can be directed to:
Mr. Mark Ball Ball.MG@forces.gc.ca 613-992-4539.

R.G. Dickinson
Director Operational Research (Joint)
for Director General Centre for Operational Research & Analysis

Enclosure: (1)

Distribution List (next page)

Distribution List

External

DRDC Valcartier Attn: DG
DRDC Atlantic Attn: DG
NORAD OR
MARLANT N02OR
MARPAC N02OR

Internal

DGJFD
DJFC
CO JIIFC Det
PD JIIFC
PM CapDEM TD (5 copies)
PM JCDS21 TD
DG CORA
DOR(Joint) + OR Teams
DOR(MLA) + OR Teams
DOR(Corp)
CANDACOM ORT
CEFCOM ORT
JSORT (Attn: M.G. Ball) (2 hard copies, 1 PDF)
JSORT (Attn: R.W. Funk) (1 hard copies, 1PDF)
ORD Library (2 hard copies, 1 PDF)
DRDC/DRDKIM3 (2 hard copies, 1 PDF)

DOCUMENT CONTROL DATA
(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared e.g. Establishment Sponsoring a contractor's report, or tasking agency, are entered in Section 8).

Centre for Operational Research & Analysis
101 Colonel By Drive
Ottawa, Ontario, K1A 0K2

2. SECURITY CLASSIFICATION (overall security classification of the document, including special warning terms if applicable)

UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title)

Simple Executable Architecture Example – Volleyball Simulation

4. AUTHORS (last name, first name, middle initial)

Ball, Mark G.

5. DATE OF PUBLICATION (month Year of Publication of document)	6a. NO OF PAGES (total containing information. Include Annexes, Appendices, etc.)	6b. NO OF REFS (total cited in document)
December 2005	24	

7. DESCRIPTIVE NOTES (the category of document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Note

8. SPONSORING activity (the name of the department project office or laboratory sponsoring the research and development. Include the address).

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written.)
10a. ORIGINATOR's document number (the official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC CORA Technical Note TN 2005-15	10b. OTHER DOCUMENT NOS. (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors: further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
( ) Distribution limited to government departments and agencies; further distribution only as approved
( ) Distribution limited to defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the test is bilingual).

The Joint Staff Operational Research Team (JSORT) is developing a State Machine model using the CORE software package to simulate the concurrent workflows that exist in net-centric operational activities. Modelling architectures in CORE takes time and experience to learn properly so a simple model of a volleyball match was used to practice building a well-known system from start to finish. This report discusses the volleyball model elements, visualization of the match's progress and then assesses how varying player abilities affects the likely outcome of the match. The final chapter discusses possible extensions to the model and details the lessons learned through this process.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

CORE
COREsim
capability engineering
architecture

www.drdc-rddc.gc.ca