

14th ICCRTS 2009
"C² and Agility"

Title of Paper:

Functional Modeling of Agile Command and Control

Topics:

5. Experimentation and Analysis

Names of Authors:

Rogier Woltjer¹, Erik Prytz², Kip Smith³

Point of Contact:

Rogier Woltjer

Cognitive Systems Engineering Laboratory (CSELAB), Div. of Human-Centered Systems (HCS)
Dept. of Computer and Information Science (IDA), Linköpings universitet
SE-581 83 Linköping, Sweden
Phone: +46 13 28 2289, E-mail Address: rogwo@ida.liu.se

Paper ID# 097_S

Abstract

A critical element to successful command and control (C²) is developing and updating an accurate and lucid model of the interdependencies between functional units, e.g., multiple platoons of artillery and tanks. Two of the challenges to this understanding are (1) the adoption of a detailed description of interdependency and the associated understanding of interdependent functions (Brehmer, 2007) and (2) the application of that description to both own and opponent forces' opportunities and vulnerabilities to provide for agility (Alberts, 2007). This paper documents a straightforward approach to modeling functional interdependency that addresses these challenges. The Functional Resonance Analysis Method (FRAM; Hollnagel, 2004) is shown to describe the C² functions of the DOODA loop (Brehmer, 2007) and the tactical and operational functions of military activity. FRAM models are applied to own and opponent forces in a computer-based dynamic war-game (DKE) to reveal and characterize both agile and unsuccessful C² practice.

¹ PhD student (Cognitive Systems), fulltime.

² BSc/MSc student (Cognitive Science), fulltime.

³ PhD (Management). Contribution less than 50% according to student paper eligibility criteria.

Functional Modeling of Agile Command and Control

Rogier Woltjer, Erik Prytz, Kip Smith

Cognitive Systems Engineering Laboratory (CSELAB), Div. of Human-Centered Systems (HCS)
Dept. of Computer and Information Science (IDA), Linköpings universitet, Linköping, Sweden

Abstract

A critical element to successful command and control (C²) is developing and updating an accurate and lucid model of the interdependencies between functional units, e.g., multiple platoons of artillery and tanks. Two of the challenges to this understanding are (1) the adoption of a detailed description of interdependency and the associated understanding of interdependent functions (Brehmer, 2007) and (2) the application of that description to both own and opponent forces' opportunities and vulnerabilities to provide for agility (Alberts, 2007). This paper documents an approach to modeling functional interdependency that addresses these challenges. The Functional Resonance Analysis Method (FRAM; Hollnagel, 2004) is shown to describe the C² functions of the DOODA loop (Brehmer, 2007) and the tactical and operational functions of military activity. FRAM models are applied to own and opponent forces in a computer-based dynamic war-game (DKE) to reveal and characterize both agile and unsuccessful C² practice.

Introduction

Command and control is in a state of change. Both civilian emergency managers and military commanders seek to implement agile network-based organizations. The drive for command and control agility stems from coordination problems in complex environments that cannot be solved by traditional hierarchic command structures. For example, civilian (emergency management) agencies want to improve inter-agency coordination (Smith & Dowell, 2000) and the military wants to overcome problems with agility, combat power and speed (Alberts, 2007; Alberts et al., 1999; Cebrowski and Garstka, 1998). This concept steps away from the traditional "platform-centric" hierarchical command structure where power is partially lost in top-down command-directed synchronization. The new approach envisions a high-performance information network that makes sensor information available to the entire command structure, characterized by agility.

"Agility ... is the synergistic combination of robustness, resilience, responsiveness, flexibility, innovation, and adaptation. Each of these attributes of agility contributes to the ability of an entity ... to be effective in the face of a dynamic situation, unexpected circumstances, or sustaining damage. Effectiveness without agility is fragility." (Alberts, 2007, p. 23).

Agile network-based command and control is about the linking or networking of well-informed units so that each of these units can take action locally on the basis of information and collaboration of others (Alberts, 2007) and to self-synchronize in order to meet a commander's intent (Brehmer and Sundin, 2000, 2005; Cebrowski and Garstka, 1998). The units of analysis for understanding and predicting the activity of agile command and control systems are, accordingly, the functions to be performed by nodes in the network, their interactions, and their interdependencies. Thus, finding appropriate couplings between functional units is crucial. Agility is, according to Alberts (2007), achieved through the units' *focus* (shared understanding and actionable intent) and *convergence* (coordinated unified movement towards a specific outcome). Furthermore, the diverse terms used in the description of agility imply the importance of (1) *anticipating* threats, expecting the unexpected and preparing for surprise, and adapting preventively, (2)

responding and adapting with quick ease and resourcefulness to expected and unexpected threats when they occur, and (3) flexibility and adaptability in *recovering* from adverse events to return to initial functioning or establish new functions. (These issues are also being discussed in the emerging field of resilience engineering, cf. Hollnagel & Woods, 2006; Hollnagel, Nemeth, & Dekker, 2008). Note that, in principle, threats and adverse events may be anything that emergency or military forces may be subjected to, from actions by an adversary, to events in natural phenomena that pose hazards and emergencies, or other perturbations in the functioning of own forces.

A key to C² agility in a military setting is thus developing and updating a viable and dynamic model that describes and predicts (a) the interdependencies among the functions (to be) carried out by own (and opposing) forces and (b) the constraints on functions in battle. Such a model should facilitate agility, e.g., be an aid to anticipate specific threats and prepare for unspecified adverse events, and establish shared understanding of current threats and viable future responses and adaptations, resulting in coordinated and informed action. The challenge to agility is exacerbated by the growing need for joint operations with a variety of civilian agencies (for example, rescue services, local and national governments, and private-sector agencies) when working to resolve emergency or conflict situations. The international nature of many emergencies, disasters, and conflicts compels civilian and military services to cooperate across traditional organizational and cultural boundaries. The prospect of having to coordinate activities across agencies, nations, and cultures suggests that a whole new range of technical and social challenges may emerge that can only be addressed by a truly agile command-and-control structure. The provision of a way of modeling critical (own and adversary) functions in past, current, and future operations may provide shared and actionable understanding of these challenges.

A very general model often adopted within the military is Boyd's Observe-Orient-Decide-Act (OODA) loop. However, the OODA loop provides few specifics on how to achieve agile C². This situation presents a pair of daunting challenges to those who seek to develop and update concise and accurate models of an agile system: (1) the adoption of a detailed description of dynamic interdependency and associated understanding of interdependent functions (Brehmer, 2007) and (2) the application of that description to both own and opponent forces' opportunities and vulnerabilities to provide for agility and resilience (Alberts, 2007).

This study applies the Functional Resonance Analysis Method (FRAM; Hollnagel, 2004) to a computer-based dynamic war-game (DKE) and documents that FRAM is a viable approach to addressing these challenges. FRAM models describe functions that are performed over time in terms of their input/output and constraints of time, resources, preconditions, and controls. A DKE experiment is analyzed where pairs of teams of three commanders play a game similar to "capture the flag". The paper illustrates how FRAM models describe the C² functions of the Dynamic-OODA (DOODA) loop (Brehmer, 2007) and the operational and tactical functions of military activity. FRAM models are applied to own and opponent forces to reveal and characterize both agile and unsuccessful team behavior and can be used to assess and predict function performance.

Background

We adhere to the modeling principles of cognitive systems engineering (CSE; Hollnagel and Woods, 1983, 2005). CSE addresses questions such as how to make use of the power of contemporary technology to facilitate human cognition, how to understand the interactions between humans and technologies, and how to aid design and evaluation of digital artifacts. Related to distributed cognition (Hollan et al., 2000) and macrocognition (Klein et al., 2003), CSE takes an ecological view regarding the importance of context

when addressing cognition. It focuses on constraints that are external to the cognitive system (present in the environment) rather than on the internal constraints (memory limitations, and so on) that are foci of the information processing perspective. CSE focuses on these external constraints that affect performance, as well as the possibilities for action that shape constraints.

The unit of analysis in cognitive systems engineering is the “joint cognitive system.” A cognitive system (Hollnagel and Woods, 2005) is a system that can control its goal-directed behavior on the basis of experience. The term “joint cognitive system” (Hollnagel and Woods, 2005) means that control is accomplished by an ensemble of cognitive systems and (physical and social) artifacts that exhibit goal-directed behavior. A battalion may therefore be modeled as a JCS, as can an assembly of chiefs of staff, or a scout with binoculars on a motorcycle. JCSs are defined by the functions they perform. In the areas of interest to cognitive systems engineering, typically one or several persons (controllers) and one or several support systems constitute a joint cognitive system which is typically engaged in some sort of process control in a complex environment.

JCSs may be defined at various levels, depending on the functions they perform. For example, an artillery unit may perform tactical functions such as moving and firing artillery, and is composed of cognitive systems (a commander and subordinates) and artifacts (the physical machinery). At a higher level, several such artillery units and other ground vehicles may form a JCS, performing operational functions such as defending a city or taking a bridgehead. The Extended Control Model (ECOM) in CSE describes the performance of JCSs with four parallel control loops. The control principle is based on the idea that behavior towards goals, and thereby control, is a combination of feedback (compensatory) and feedforward (anticipatory) control. At the most compensatory level, the tracking loop performs immediate feedback control with a very short time cycle. The regulating loop is a combination of feedback and feedforward control at the short term. The regulating loop provides goals and criteria to the tracking loop. The monitoring loop provides feedback testing if goals are met, has a relatively long duration time, and provides the regulating loop with plans and objectives as feedforward. Finally, the targeting loop is a feedforward loop that sets goals that form the input for the monitoring loop. The ECOM layers have similarities to the tactical, operational, and strategic levels of military command and control in their different time horizons associated to different goals and concurrency of performance.

A related approach is “dynamic decision making” (DDM) (Brehmer, 1992). This approach focuses on the functions served by decision making in order to gain control (e.g., achieve some desired state of affairs) rather than on the decisions themselves. As its name implies, dynamic decision making involves tasks with a dynamic character. This implies that a series of interdependent decisions in real time is required to reach the goal, and that the state of the decision problem changes both autonomously and as a consequence of the decision-maker’s actions. This approach describes decision making as the intersection of two control processes: (1) the system that the decision maker aims to control and (2) the means for achieving control. The second process is used to control the first (Brehmer and Allard, 1991). In a more detailed DDM modeling effort, Brehmer (2006) identifies two major shortcomings of the OODA loop: (1) the absence of a representation of the effects of the ACT stage, rendering representation of delays in C² impossible, and (2) the lack of detail in its description of the requisite functions for effective C². Brehmer further identifies two major shortcomings of existing cybernetic models of C² (which he claims also apply to the OODA loop): (1) their depiction of the C² process as an essentially reactive process, leaving no room for initiative, and (2) their generality. The latter generates a poor representation of the requisite functions for effective C². Brehmer (2005, 2006, 2007) introduced the DOODA loop to overcome these problems. The DOODA loop is a contextualized model of this control loop approach, integrating a loop of sensemaking, decision making, action, and information collection, with more military-specific concepts such as command, order, mission, and sensors, into a comprehensive model of command and control. The

DOODA loop specifies three command and control functions: Data collection, Sensemaking, and Planning. Sensors provide data about what happens in the battle space, which is collected and made sense of in an iterative process, and sensemaking precedes planning, which results in orders. These are communicated and result in military activity, which, through frictions (Von Clausewitz, 1832), leads to effects, and new data about the battle space. Brehmer's (2007) standpoint is that the way to agile command and control is through understanding the functions of C².

We use the Functional Resonance Analysis Method (FRAM; Hollnagel, 2004) to develop a description of the functions of C². FRAM is not only part of the methods used in CSE, but fits within the emerging discipline of resilience engineering (Hollnagel, Woods, & Leveson, 2006, Woltjer, 2008). Resilience engineering has clear similarities to designing for agile systems in network-based command and control, making FRAM a natural candidate for application in this domain. FRAM constructs models of the joint cognitive system's process control task by connecting modules that represent the functions that need to be performed. All functions are described through and may be linked via six aspects (abbreviated with initials); Input, Output, Precondition, Resource, Time, and Control, and are represented using the hexagonal module shown in Figure 1. The modules of functions and their characteristics are linked together to construct models of complex joint human-machine system behavior such as the tasks and functions performed in network-based command and control.

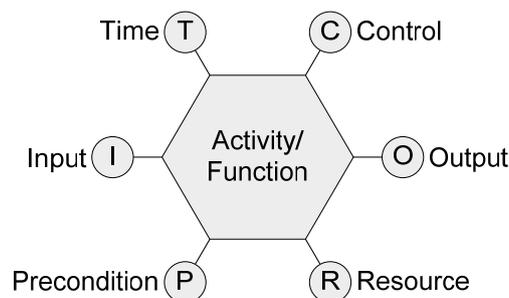


Figure 1 A FRAM function representation.

In adversarial command and control, a task force's functions ultimately support the overarching goals of (a) constraining the adversary - limiting the opponent forces and hindering them from achieving their goals, and (b) enabling behavior that is beneficial for achieving a task force's own objectives.

These considerations raise and identify a central concept of this work, *constraint*. Constraints are invariably described as essential factors in the functioning of (cognitive) systems. The salience and significance of constraints are recognized by a variety of disciplines, such as cybernetics, (general) systems theory, ecological psychology, cognitive work analysis, information processing psychology, and representation design. Combining these perspectives we define constraint as an actual and/or perceived feature of a specific system or its environment that defines limits and/or opportunities for that system's function performance (see Woltjer, 2009; Woltjer, Smith, & Hollnagel, 2008). This definition stresses both the limiting and enabling aspects of constraints. The FRAM representation makes it possible to address constraints explicitly by analyzing the behavior of joint cognitive systems in terms of function aspects. This resonates well with CSE, where constraints are said to shape the selection of appropriate action (Hollnagel and Woods, 2005).

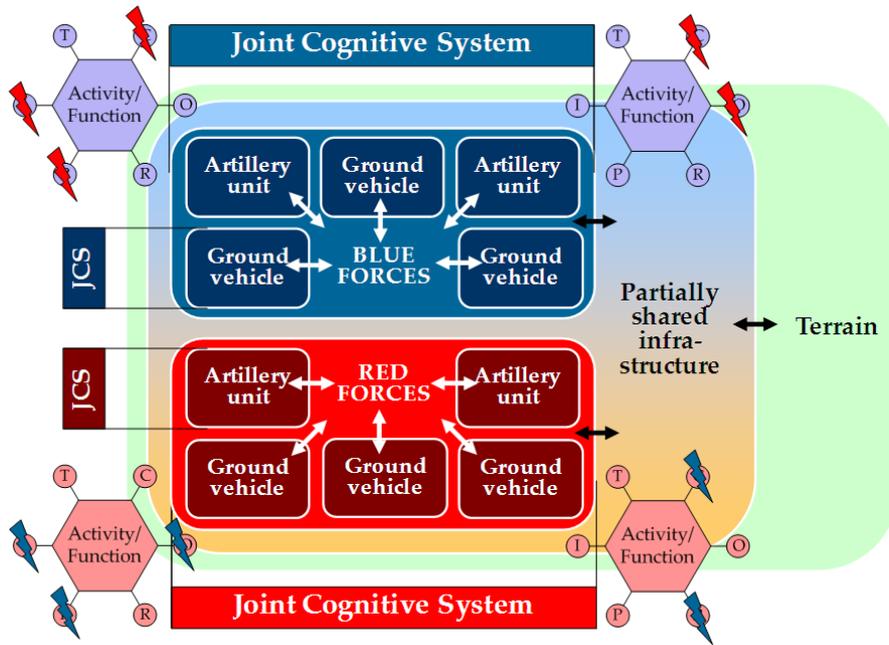


Figure 2 Joint cognitive systems (JCSs) perform functions while attempting to constrain the adversary's function performance.

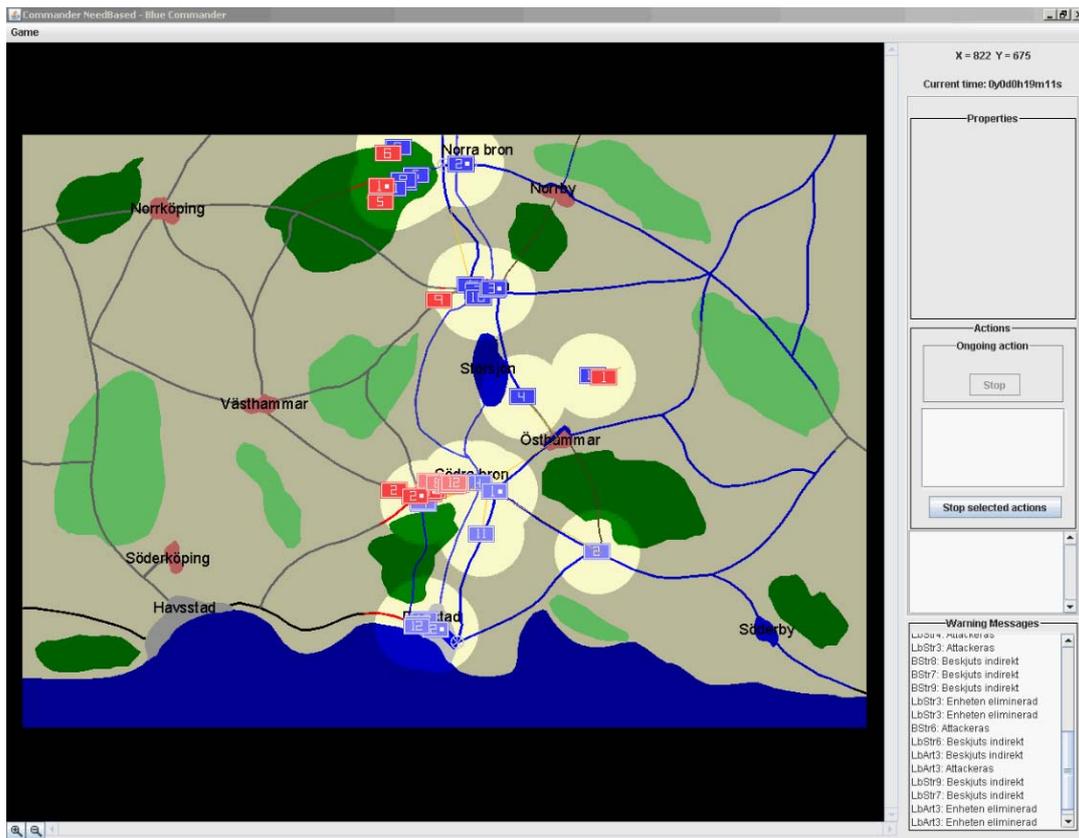


Figure 3 Blue (right) side commander view of the interface (commanders see only their own side's units and objects within visibility range. The map (terrain and vegetation) can be seen by all.

Figure 2 sketches a description of how joint cognitive systems attempt to constrain their adversary's functions in an adversarial C² setting. The FRAM representation prescribes how to develop and adapt courses of action in an efficient manner: agile C² focuses on affecting the adversary's function aspects so that their outputs cannot be realized or only outputs beneficial for own objectives are made possible. This focus is represented by the thunderbolts in Figure 2, using the opponent's colors for indicating constraint on adversary function aspects. Further, the FRAM representation informs how agile C² can translate these targeted strikes into orders. The thunderbolts specify the effects of operational and tactical functions that constrain adversary function performance, so that the adversary's objectives cannot be reached but allied objectives are made possible.

Method

To illustrate the utility of the FRAM representation and method in an adversarial C² setting, we applied it to exercises conducted using the dynamic war-game for experiments (DKE). This section first describes the DKE simulation environment, and then the analysis method.

DKE (Dynamisk Krigsspel för Experiment, Dynamic War-game for Experiments) is a war-game developed at Försvarshögskolan (FHS; the Swedish National Defence College). There are two adversaries (the blue team and the red team) who wage battle in a simulated battle environment. The DKE research team⁴ at FHS has recently conducted a study with DKE in which two teams of three people (one commander and two subordinates) played against each other.

The subordinates in each team maneuvered the units, while the commanders could only monitor the game and give orders. All players had access to a computer interface. A view from DKE combining what each player sees is presented in Figure 3. Only the own color (red, light red, blue, light blue) units are visible to the subordinates (two in each team). The lighter colored circles demarcate the visibility range of the units. Thus, all units other than the own color could only be seen by the players if they were within this visibility range. The map contains cities, different kinds of vegetation, non-traversable water; and roads, four (double) bridgeheads, and headquarters (one for each team). Various types of terrain allow for different rates of travel, defense, and protection. The objective of the game in its current configuration is to conquer (meaning to advance beyond) at least one bridge and secure a road to the own headquarters from the conquered bridge. A road is secured by moving a unit on it.

Participants were eight pairs of three-person teams none of whom had received formal training in command and control. A red and a light-red force battled a blue and light-blue force. All team members could communicate with each other by talking aloud in the room while their voices were recorded through a headset, but they were visually separated from each other. The data collected consists of replayable log files of movements and actions of units, and audio recordings of each of the team members individually. The analysis of a subset of data in the log files led to a description of the functions that the analyzed teams in the study performed, and to a characterization of constraints in the form of FRAM function aspects.

To find the FRAM-modules, one may start with the top-level goal, which may translate into the top-level function, or one may start with any function and move on to related functions. Typically, the goal in command and control is to get some sort of process under control, such as a progressing fire, a crowd of panicking people, a spreading oil spill, or a moving adversarial military force. This translates into the top-level function of, for example, fighting the fire, directing the flow of people, blocking and cleaning the oil spill, or defeating the adversary. From there on, the functions that are necessary or otherwise related to

⁴ See Acknowledgments.

the input of the function — its timing, control, required resources, and preconditions — determine the other functions in the model of a task.

Functions are presented in the format outlined in Table 1, and visualized by using the hexagonal representation illustrated in Figure 1.

Subsequent steps in FRAM continue by describing function performance, identifying dependencies/couplings among functions (Hollnagel, 2004). The output of the functional description is a list of functions each with their six aspects. Functions may be coupled via their aspects. For example, the output of one function may be an input to another function, or provide a resource, fulfill a pre-condition, or enforce a control or time constraint. The couplings between functions are found by analyzing and identifying common or related aspects.

Table 1 Function description template

Function Name	Description	Essential Variables
Input	What is transformed during function performance to produce the output	Essential variables that describe the input
Output	What is produced by a function, described as a changed state after <i>successful</i> function performance	Essential variables that change state when output is produced
Preconditions	Conditions (states) that must be fulfilled at the start of or during function performance	Essential variables that describe preconditions
Resources	What is needed and used to process input	Essential variables that change state as resources are used
Time	Time aspects, including time needed and time available to perform a function	Essential variables that describe time aspects
Control	What supervises, steers, controls, plans, or adjusts function performance, such as deciding when a function should be performed	Essential variables that describe controls

Results

Building on our analysis of the DKE environment, on the levels of military activity, and on Brehmer's (2007) DOODA loop, three categories⁵ of functions can be identified. (1) Tactical functions (e.g., move, fight) are performed by the units in the simulation, and are observable when data is available. (2) Operational functions (e.g., take bridgehead, secure road) are inferred and ascribed to groups of units that aim to reach operational objectives and consist of various performances of tactical functions. (3) C² functions (data collection, sensemaking, and planning, according to Brehmer, 2007) are general functions that enable the exercise of command and control. C² functions are not directly observable but may be inferred, by studying groups of units and the behavior of the commanders that comprise the C² system (e.g., retrieving data, giving orders, discussing tactics).

⁵ More categories of functions could be identified, such as strategic functions. Strategies are plans for high-level goals that specify operations of the entire team during (large parts of) an entire trial. Strategic functions would be a collection of operational functions with a common/joint goal (output). As strategic functions are merely another layer above operational functions in the same sense that operational supersede tactical functions, strategic functions are not treated here and do not change the modeling effort in a conceptual or substantial way.

Tactical functions

Four different Tactical functions were identified: Move, Fight, Artillery Fire and Manage Resources. The functions Move and Fight are explained in some detail to exemplify the FRAM way of describing functions. Table 2 summarizes the description of the Move function in terms of the FRAM function aspects and their essential variables.

Table 2 Move function

Move	Description	Essential Variables
Input	OWNFOR Unit	OWNFOR.Unit.Position, OWNFOR.Unit.TargetPosition
Output	<p><i>WHILE Active :</i> OWNFOR Unit's TacticalStatus is Moving OWNFOR Unit Position's Owner is OWNFOR OWNFOR Unit Position is changing</p> <p><i>AT FinishingTime :</i> OWNFOR Unit's Position has changed OWNFOR Unit Positions' Owner is OWNFOR</p>	Coordinate().Owner OWNFOR.Unit.Position and .TacticalStatus
Preconditions	OWNFOR Unit's TacticalStatus is not Fighting OWNFOR Unit's TacticalStatus is not Firing Artillery	OWNFOR.Unit.TacticalStatus
Resources	OWNFOR Unit's Fuel	OWNFOR.Unit.FuelLevel
Time	<p><i>Performance Time is a function of</i> OWNFOR Unit's Movement Type, OWNFOR Unit Position's Terrain Type, OWNFOR Unit's Fuel Level, and OWNFOR Unit Fuel Level's Speed Constant</p>	OWNFOR.Unit.MovementType, Coordinate().TerrainType, OWNFOR.Unit.FuelLevel
Control	Subordinate	OWNFOR.RoleAllocation

The Move function takes one unit as input, and while the function is active that unit will move towards a specified target position. When a unit moves over a road, city or bridgehead position, that particular position changes owner to the owner of the unit. Units can be instructed to move following roads, or travel in a straight line through vegetation. The function terminates when the unit has reached the target position or when one of the preconditions are no longer met. The preconditions for the Move function is that the unit cannot be engaged in the Fight or Artillery Fire function. It is thus possible to make the opponent's unit stop by attacking it and therefore forcing it to engage in the Fight function. The resource consumed during Move is Fuel. Low fuel level slows the unit down (but doesn't stop it). The speed of the unit is also affected by the type of terrain the unit is moving through (e.g., vegetation is slower than road, units stop in water).

Fight is a tactical function that has a major impact on all the other Tactical functions. While a unit is engaged in the Fight function it cannot perform any of the other Tactical functions, with the exception of Artillery Fire. The input to the Fight function is one of the own units and one of the opponents units. To engage in a Fight, either unit simply has to move within the fire range (marked by a green circle centered on the unit) with regards to the other unit. There are four possible outcome states for each unit performing the Fight function; Unchanged, Retreat, Loss or Destroyed. Destroyed means that the unit is destroyed and cannot be used in further game play. The other three possible outcomes have various effects on resources such as Armor and Position. The outcome state is decided by a combination of these resources and a random number. The more units one side is engaging in a fight, the more likely it is that the Fight will have a positive outcome for them and a negative one for the opponents. A unit can become

Disturbed for 50 seconds after being fired upon with Artillery, which affects the outcome of a Fight. Fights are initiated by a subordinate within the team of commanders, or through an “Auto Defense” mechanism. Table 3 summarizes the description of the Fight function in terms of the FRAM function aspects and their essential variables.

Table 3 Fight function

Fight	Description	Essential Variables
Input	OWNFOR Units OPPFOR Units	OWNFOR.Unit.Position, OPPFOR.Unit.Position
Output	<i>WHILE Active</i> : OWNFOR Unit’s TacticalStatus is Fight, <i>AT FinishingTime</i> : OWNFOR’s and OPFFOR’s Units’ Armor, Position, Stamina, Fuel, Ammunition, Attack Strength, have possibly changed and their Status is possibly Destroyed	OWNFOR/OPPFOR.Units .ArmorLevel, .Position, .Stamina, .FuelLevel, .AmmunitionLevel, .AttackStrength, .TacticalStatus
Preconditions	<i>AT Starting Time & WHILE Active</i> : Distance between OWNFOR and OPFFOR Unit Positions is within Firerange	OWNFOR.Unit.Positions, OPPFOR.Unit.Positions, OWNFOR.Unit.FireRange, OPPFOR.Unit.FireRange
Resources	OWNFOR Unit’s Ammunition, Fuel, Stamina, Armor, Attack Strength	OWNFOR.Unit.AmmunitionLevel, .FuelLevel, .ArmorLevel, .Stamina, .AttackStrength
Time	<i>Performance Time</i> is Fight Duration	FightDuration ≤ 30 sec.
Control	Subordinate, AutoDefense	OWNFOR.RoleAllocation

Artillery Fire is a function that only can be used by Artillery units. It is used to fire on opposing units from a longer distance to weaken them by reducing the variables Armor, Attack Strength and Stamina or by inflicting the status Disturbed on that unit. A third possible outcome state is Unchanged, in which the unit fired upon is not affected. The Artillery function is quite similar to Fight, but entails a different fire range, duration, effects, and movement possibilities. Several artillery units may be combined for higher damage, and several target units can be affected by the one artillery unit. “Automatic Fire” is available similar to “Auto Defense” in Fight⁶.

Manage Resources is a function performed to regain essential variables Fuel, Ammunition, Attack strength, Armor and Stamina. While a unit performs Manage Resources it cannot move or Fight and if an enemy unit actively engages the unit through Fight it will cause the Manage Resources function to end without any recovery for the unit.

Given this description of the four tactical functions, interactions between the various tactical functions may thus be identified. This is done by examining the potential couplings between the six aspects of the functions. Thus, a logical structure of how the tactical functions may be interrelated is obtained, as in Figure 4. Some of the connections are described here in further detail, using the function names and first capital letters of function aspects Intput, Output, Precondition, Resource, Time, and Control:

⁶ Note that, due to the recursive manner in which functions are defined in FRAM, the Automatic Fire and Auto Defense abilities may be modeled as sub-functions of Artillery Fire and Fight, respectively. Here the analysis does not call for such a division, but FRAM allows functions to be split up if the purpose of analysis necessitates this.

For each side, an own Move O (moving within fire range) enables a Fight P (being within fire range), as well as an own Move O (moving beyond fire range) disables a Fight P (being within fire range). This holds also for one side's Move O (moving within/beyond fire range) and the other side's Fight P and Artillery Fire P, although these fire ranges are different. Move O becomes Fight I as units first move and then fight. Fight O reduces the fuel resource R for a later Move. Once fighting, both Fight functions' I's are identical, since blue and red Fights transform the same units, on either side. These functions also match P's since both are within identical fire range when fighting. The performance time T of both Fights are also related.

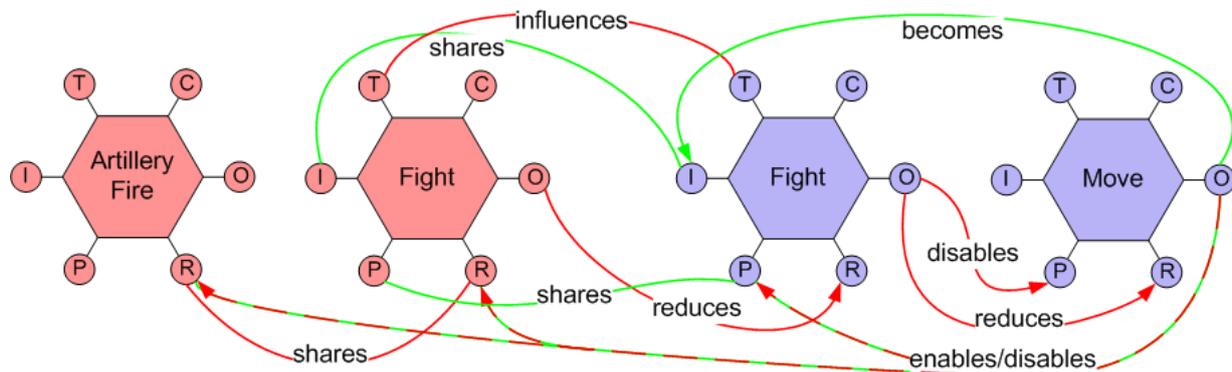


Figure 4 Interdependencies between tactical functions of both sides (two on left-side red, two on right-side blue). Relations are bidirectional unless specified otherwise by a unidirectional arrow. Red (dark) links indicate the *limiting* aspect of constraint, green (light) links the *enabling* aspect.

Operational functions

Nine Operational functions were identified: Take (obtaining “ownership” over an object), Keep (maintaining ownership), Secure Road (a Take of a road), Join Forces (moving units to reinforce other units), Bypass (going behind enemy lines), Place Blockade (placing units to block the adversary), Raid (undoing the adversary’s securing of roads by crossing them), Hunt (going after an adversary that has succeeded to Bypass), Scout (moving a unit in order to extend the viewable area), and Retreat (withdrawing from an object or area towards own headquarters in the face of adversary threat).

The Take <Object> function’s output is that the side performing the function has control over all positions that define the Object. When a team is moving one or several units to capture an object (a bridgehead or city), with or without fighting enemy units for control, they are performing a Take function. Operational functions are decided upon by the command team, therefore the control is in the hands of commander and subordinate.

Operational and tactical functions are related logically in the FRAM description of DKE. An operational function, such as Take Bridgehead, may be performed through performing a number of its tactical sub-functions, numerous times, sequentially or concurrently. The description of operational or tactical functions is a choice of granularity that should be made by keeping the purpose of the description in mind. The operational and tactical levels of functions are thus different levels of description of the same military activity.

Figure 5 illustrates how operational functions may constrain (limit and enable) each other in a network of interdependent functions. Some of the connections are described here in further detail, using the function names and first capital letters of function aspects Input, Output, Precondition, Resource, Time, and Control.

Table 4 Take function

Take <Object>	Description	Essential Variables
Input	OWNFOR Units, Object	Object.Type
Output	<p><i>WHILE Active :</i> OWNFOR Unit's Operational Status is Take(Object)</p> <p><i>AT FinishingTime :</i> All Positions' Owner in Object is OWNFOR</p>	<p>OWNFOR.OperationalStatus,</p> <p>Coordinate().Owner</p>
Preconditions	<p><i>AT StartingTime :</i> Not all Positions' Owner in Object is OWNFOR</p>	Coordinate(Position).Owner
Resources	OWNFOR Units	
Time	<p><i>Available Time is (Game Time – (Starting Time + Performance Time)),</i></p> <p><i>Performance Time is a function of all Units' Move, Fight, Manage Resources, ArtilleryFire Performance Times</i></p>	<p>GameTime ≤ 30 min,</p> <p>StartingTime,</p> <p>PerformanceTimes</p>
Control	Commander, Subordinate	
Sub-functions	Move, Fight, Manage Resources, ArtilleryFire	

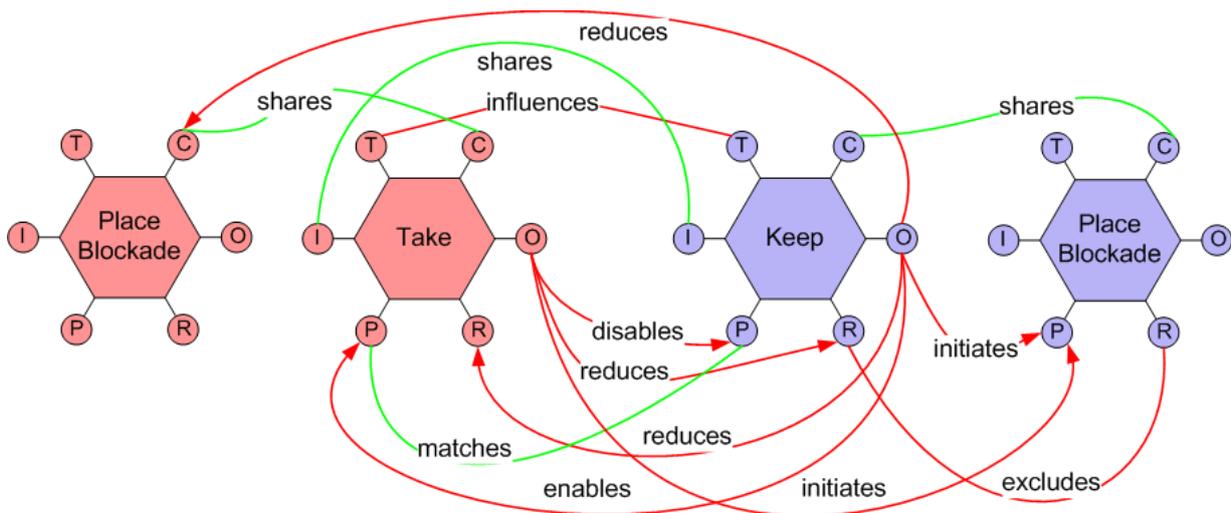


Figure 5 Interdependencies between operational functions of both sides (two on left-side red, two on right-side blue). Relations are bidirectional unless specified otherwise by a unidirectional arrow. Red (dark) links indicate the *limiting* aspect of constraint, green (light) links the *enabling* aspect.

Take I (units, object) shares Keep I (units, object): This means that the two functions may take the same object as input, and that the same units may be involved. For example, blue units fight red units to keep a bridgehead, while the same red units fight the blue units to take the same bridgehead.

Take P (one side not owning the object) matches Keep P (the other side owning the object): This connection, along with the previous, indicates that the Take and Keep functions can be performed simultaneously (and on the same object). Similarly, Take O (one side owning the object) disables Keep P (the other side also owning the object). Combining the last two relations, Keep O (one side owning the object) enables Take P (the other side not owning the object). However, not always all of these relations are valid, since one side can perform a Take without that the other side is already present to perform a

Keep. The applicability of these relations (links) depends on actual function performance (see the Example instantiation below).

Keep O (one side owning the object) reduces Take R (the other side's units and their resources), through fighting performed in keep and take on both sides.

Take O (one side gaining ownership of the object) and failure of Keep O (the other side losing ownership of the object) may initiate Place Blockade P: If the Keep function fails, or Take functions succeeds depending on your point of view, the Place Blockade is active. It is now what is preventing enemy units from advancing further. This assumes that the blue Place Blockade is located close (geographically) to the position where the Take and Keep functions are performed (indicating that links may be conditional and not always valid).

Red Take C shares red Place Blockade C, and blue Keep C shares blue Place Blockade C, indicating that the same commander may control both functions. While this increases flexibility and decreases communication demands, it increases the commander's attention demands. Thus, this link may have positive and negative effects. Similarly, blue Keep O may reduce red Place Blockade C, since one side performing a Keep may distract the other side's geographically distant activities.

Keep T and Take T influence one another: The time to execute a function is a union of all sub-functions' performance times. These performance times vary depending on the adversary's performance.

Keep R excludes Place Blockade R and vice versa: Each side has a limited number of units, so assigning a unit to either function excludes that unit from performing the other function.

These abstract functions take on specific values as they are being performed. In FRAM, descriptions of the performance functions at a certain time (interval) are called instantiations of functions. The following example will discuss several instantiations during a specific period in one of the trials recorded during the experiment.

Example instantiations

In the example scenario, the red and blue sides employed different tactics. While the blue side moved quickly to occupy all four bridges, red focused on amassing a large number of units close to one of the bridges, Södra Bron, while sending few if any units to the other bridges. The battle that followed at Södra Bron is presented here in detail to illustrate tactical, operational, and command and control functions and relationships between these classes of functions.

Tactical and operational functions

In the twelfth minute of the scenario, the red units⁷ west of Södra Bron, LrStr1-10 & 12 and LrArt2, started to move toward Södra Bron in order to take control of it. This is an example of a Take Object function. They were using the tactical function Move in order to realize Take Södra Bron. However, they were not moving straight toward the bridge, but rather used several short moves back and forth with all units which gradually took them closer to the bridge. All of these Move function instantiations jointly describe the Take Södra Bron operational function instantiation. In response to the red activity, three Light Blue units were sent from Flodstad as reinforcements, by performing the Join Forces function using the tactical function Move. LbArt1 moved along the east side of the river while LbStr7 and 9 moved on the west side,

⁷ Units identifiers used here are composed of three parts: *Color* (B for Blue, Lb for Light blue, R for Red, Lr for Light red), *unit type* (Str for tank, Art for artillery), and identifying *number*.

thus emerging behind the red forces. As the red units drew closer, LbStr1 was also sent toward Södra Bron and LbArt3 was again pulled back behind the other forces to the east side of Södra Bron.

The excerpt in Table 5 shows some aspects of some of the Move-functions that occurred at this point and which operational function these constituted. The light red units (e.g. LrStr10) used short moves, only a few seconds long, while the light blue units (e.g. LbStr9) used move functions that took minutes to cover greater distances. The long move was more efficient as it minimizes the downtime and allows the operator to focus on other functions while the units are moving. As can be seen in the precondition (Prec.) column for LbStr9 & 10, the Move function ended with these units fighting other units; since the units cannot move and fight simultaneously, the precondition for Move is unfulfilled. The fight function they initiated is shown later. For the Input, the unit ID is shown, for Time the starting and ending time of the tactical function performance, for Output the coordinates of the units at the end time, for Resources the fuel level at start and end time.

Table 5 Tactical (Move) and operational function instantiation examples, light blue tanks 5, 7, 9, 10.

Tactical Function (TF)	Input Unit	Time TF Start	Time TF End	Output Coordinates	Prec.	Res. Fuel Start	Res. Fuel End	Operational Function
Move	LbStr9	12:10	16:21	X471 Y436	Fight	70	52	Join Forces
Move	LrStr10	12:11	12:24	X481 Y427		41	40	Take Södra Bron
Move	LbStr7	12:17	16:27	X499 Y435	Fight	72	54	Join Forces
Move	LrStr5	12:24	12:34	X474 Y428		56	55	Take Södra Bron
Move	LrStr10	12:27	12:32	X474 Y428		40	39	Take Södra Bron

Figure 6 shows a close-up of Södra Bron at time 14:26. Several red units (not all visible as they cover each other) were facing a few blue units. LbStr9 & 7 (7 “under” 9) were advancing to attack the red units from the south. The pattern of short movements of red units continued, and blue sent more and more units to reinforce the bridge. Figure 7 shows the operational functions that are performed at the time of the screenshot, see the rightmost column in Table 5.

Operational, tactical, and opponent’s tactical functions

At 15:15, LbArt3, an artillery⁸ unit, fired at the red units. Table 6 describes three of the twelve red units that were hit by this salvo. LrStr8 both was disturbed (reducing the unit’s ability to fight for 50 seconds) and lost 1 point of armor, stamina and attack. LrStr7 was only disturbed and LrStr6 was unaffected by the artillery.

Table 6 Artillery Fire function example, light red artillery unit 3, from time 15:15.

Unit	Operational Function	Tactical Function	Target	Target state change	Armor (change)	Stamina (change)	Attack (change)
LbArt3	Keep Södra Bron	Artillery Fire	LrStr8	Disturbed, Loss	5 (-1)	5 (-1)	9 (-1)
LbArt3	Keep Södra Bron	Artillery Fire	LrStr7	Disturbed	6 (0)	6 (0)	10 (0)
LbArt3	Keep Södra Bron	Artillery Fire	LrStr6	Unchanged	6 (0)	6 (0)	10 (0)

⁸ In the screenshots of DKE, artillery units are indicated by their number and a block, e.g. 3□. Tanks are indicated only by their number.



Figure 6 Screenshot close-up of Södra Bron at time 14:26. Red artillery 2 and light red tank 12, light blue tank 9, artillery 3, and tank 6, are visible.

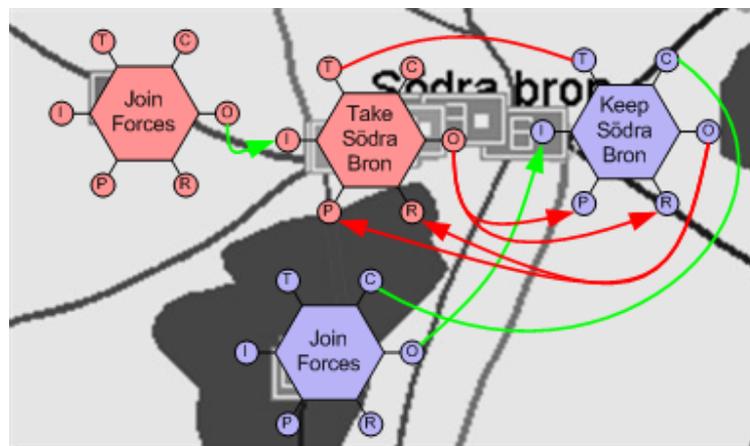


Figure 7 Screenshot close-up of Södra Bron at time 14:26, with superimposed the operational functions. Limiting (red, dark) and enabling (green, light) interdependencies between functions are indicated. Red artillery 2 performs Join Forces, tanks around light red tank 12 Take Södra Bron, light blue tank 9 Join Forces, and the group containing light blue artillery 3 and tank 6 Keep Södra Bron.

The Light Blue units on the western side of Södra Bron, LbStr4 & 5, met the advancing red units and several instances of the Fight function occurred during the following minutes (starting at 16:01). The blue units, often outnumbered 1 to 8, were forced to retreat or be destroyed. The red side did not advance fast despite this, as they continued their small moves to the east. As the red side moved slowly, the Light Blue commander sent his own units to counterattack them, despite the disadvantage in numbers, in order to slow them down further. LbStr2 & 11 were also sent as additional reinforcements as well as BStr4. LbStr7 & 9 encountered only RStr12 and RArt2 & 3 as they were attacking the rear of the red forces, but still lost their Fights and were destroyed. At about 20 minutes into the scenario, LrStr1-10 & 12, RStr2, 4 & 11 and RArt2 & 3 (16 units) faced LbStr1, 4, 6 & 7 and LbArt1 (5 units) at Södra Bron.

Table 7 shows some of the fight functions initiated during the battle of Södra Bron. Blue units were attempting to keep control of the bridge (i.e. performing the Keep function), while the red units were trying to gain control (i.e. performing the Take function). This example illustrates the interaction and the attempted exercise of constraints of both sides on the other's functions. It also shows the intermingled fashion in which tactical functions attempt to constrain opponent tactical functions: LrArt2 attacks LbStr6, but LbStr6 in turn attacks many other units besides LrArt2.

Table 7 Some of the fight functions during the battle of Södra Bron, from time 18:37.

Time Start	Time End	Unit	Operational Function	Tactical Function	Opponent
18:37	19:07	LrArt2	Take Södra Bron	Fight	LbStr6
18:37	19:07	LbStr1	Keep Södra Bron	Fight	LrArt2, LrStr1, LrStr3, LrStr12
18:40	19:10	LrStr9	Take Södra Bron	Fight	LbArt3
18:40	19:10	LbArt3	Keep Södra Bron	Fight	LrStr5, LrStr9
18:41	19:11	LbStr6	Keep Södra Bron	Fight	LrStr1, LrStr8, LrStr10, LrStr6, LrArt2
18:50	19:20	RArt2	Take Södra Bron	Fight	LbStr9, LbStr7

Figure 8 shows links between operational functions of both sides imposed on a screenshot of the game at time 19:11. Blue had control of all bridges (using various instantiations of the Keep function) and red was focusing on a single bridge with a Take function. The red Take O was connected to the blue Keep P, and vice versa, as described above. Blue was sending more units to that bridge using the Join Forces (JF) function. The JF functions are connected with an enabling link to the Take functions which these units are aimed to support, and with a limiting link to the Keep functions which they are sent from. At the top of the image, blue is trying to force its way through the red units using a Bypass versus the red Place Blockade. A red unit had slipped past the majority of the blue forces using Bypass, but was hunted by a blue unit. The red Place Blockade (PB) in the lower left may seem redundant (as well as the blue Place Blockade in the lower right), as no units are attempting to Bypass. However, since the operators' field of vision is limited, they cannot know for certain if there is an enemy unit attempting to Bypass or not, and the commanders thus prepare for this possibility.

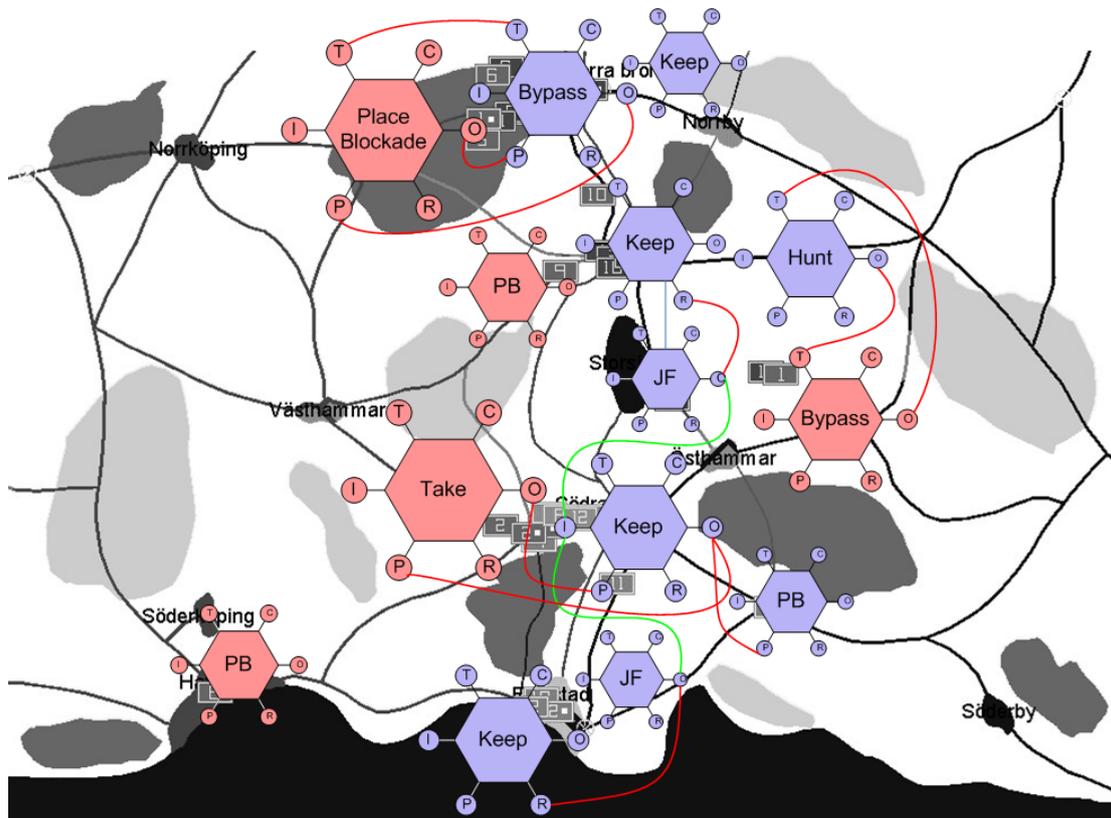


Figure 8 Examples of links between operational functions, at time 19:11.

C² functions

C² functions are the functions that a JCS of commanders and analysts and their technical support systems, together called the C² system, performs, as well as each of the team members individually. The C² system is embedded in a mission loop that connects C² and the battlefield environment. Brehmer’s DOODA loop describes three C² functions: Data collection, Sensemaking, and Planning. Sensors provide data about what happens in the battle space, which is collected and made sense of in an iterative process. The Sensemaking function produces an action-oriented understanding of the mission and the situation at hand in the form of a Course of Action (COA), and receives inputs from the specification of the mission and the data acquired by the Data collection function. The Planning function translates the COA into orders, which are the most important output of the C² system. Orders are translated into military action, which may here be described as the operational and tactical functions as modeled above. These actions are then filtered through frictions and result in effects, which are picked up through sensors by the Data collection function. Figure 9 sketches a FRAM representation of the DOODA loop.

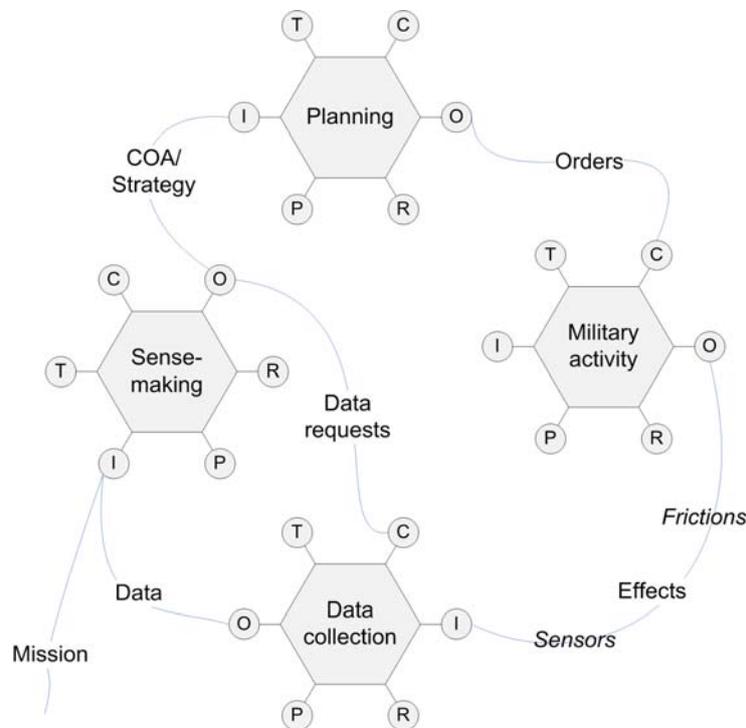


Figure 9 Potential couplings between C² functions, and military activity.

The relations between Data collection, Sensemaking, and Planning are to be seen as logical relations in the form of input and output that are continuously updated, rather than causal or temporal relations. Figure 8 sketches how the operational and tactical functions are linked logically in a FRAM description of DKE. In FRAM terms, Orders are controls on the military activity. Here we can recognize the hierarchical way of relaying orders, from the C² system’s strategy and their orders, as a control to the operational level, which in turn controls the Tactical level. The operational and tactical functions are thus different ways of description, with different granularity, of military activity. Depending on the perspective of the analysis different levels can be chosen to describe the same military activity. In agile network-based command and control, however, functions are not only performed on the basis of higher-level orders, but also as initiated at the operational and tactical levels themselves, based on communicated intent or other forms of orders. This corresponds to Brehmer’s description that DOODA loops are performed at each of these

levels of military activity, all involving different joint cognitive systems working at different time horizons.

Like operational functions, C² functions cannot be observed directly from the data describing the activities of units in the simulation. C² functions must be inferred. Combining communication transcripts with FRAM instantiations, C² functions and links between C², operational and tactical functions may be analyzed, as in the examples below. The descriptions refer to the situations depicted in Figures 3 and 8.

Example instantiations

An examples of the planning function resulting in orders is given in the following dialogue between the blue commander and subordinate:

A: Can I go and help?

C: No. I thought, what you can do, you pull up nine and seven. Take nine and seven from below Flodstad and go to Södra Bron with them.

A: Yes.

The blue commander (C) here planned and gave the order to A to perform the action that would somewhat later lead to the screenshot in Figure 6. Around that time, when tanks 7 and 9 were joining forces at Södra Bron (blue 9 is shown at the bottom of Figure 6), there is some confusion.

A: ... But look nine and them have started to freak out.

C: No, they're on their way to where they should go.

We interpret this as communication that shows sensemaking by the team, and at the same time this example sketches the differences between the DOODA loops performed in parallel by all team members. C had the idea of 7 and 9 joining forces at Södra Bron, as the order above showed, but A had not understood or recalled that order. Thus, the sensemaking function of both did not match in their output before this communication takes place. Note also the difference in orders in the DOODA loop between C and A. An order for C means instructing A to move 7 and 9, which A needs to transform into an instruction to tanks 7 and 9 to start to move accordingly. Due to time delays, these functions may differ considerably, leading to suboptimal performance and/or confusion during communication.

Not only the sensemaking and planning functions are shared, also data collection functions are shared as the following excerpt demonstrates. After some ambiguity regarding C's order to A, C realized that A has the wrong zoom for A to understand the order, so that C issued instructions regarding where to zoom, and thus which data to collect.

C: Your number two should where number one is and number one should go to Södra Bron.

A: So number two should back up?

C: No, number two should go forward. That is tank two which is in Söderby.

A: I'm in Flodstad me.

C: But look at Söderby, zoom out. Zoom out so that you see the whole map.

When the red side lose their last unit at Flodstad in a fight with four blue units, the red commander (C) reacts by saying that "red eight" (RStr8) should continue moving from its current position and place a blockade in Havsstad. This in order to prevent the blue units from raiding the roads controlled by red. C says "hope that the blue hasn't driven into anything" and "they can just drive the highway to Havsstad" meaning that he can see how the blue side now has an opportunity to raid their roads, and hope that RStr8 can

reach Havsstad in time to prevent this from happening. In other words, C tried to make sense of the new situation that has arisen from the frictions on the battlefield (i.e. sensemaking), planned a response (protecting the roads by placing a blockade with RStr8 in Havsstad) and communicated this as an order to subordinate B. B executed the order of placing a blockade (an operational function) in Havsstad by conducting the tactical function Move on RStr8.

C: Uh oh. Now it would be good if red eight, furthest west there, drives down pretty fast to Havsstad and hope that the blue hasn't driven into anything.

B: What did you say now?

C: That red eight, furthest west. It should continue down to Havsstad.

B: Yes, it's already on its way.

C: Because the blue just knocked out our last in Flodstad and then they can just drive the highway to Havsstad.

In another example, Blue unit 6 was attempting to bypass at the very top of the scenario map, above the red units that have gathered in the forest outside of Norra Bron. However, as RStr6 was moving straight toward BStr6, BStr6 aborted its mission and moves back towards the own lines. However, as illustrated by the dialogue below, B did not see where BStr6 moved to. C and B were trying to make sense of the situation and discussing whether it would be possible for the blue unit to move out of their field of vision to the west, which would place it behind their lines, or not.

B: Now I wasn't able to see what happened with that guy up there, to the north. Him six or whatever it is.

C: Well, he can't have passed anyway.

B: Are you sure?

C: Yes, quite.

B: I was busy down there.

C: It can't have moved that fast

B: I go here just to check. No, it couldn't have.

As seen in the excerpt, RStr6 was moving up to block BStr6 from passing. At 14:57, it has completed its tactical function Move and thus also completed the operational function Place blockade". However, as there were doubts whether BStr6 already has passed the blockade or not (if it had, the blockade would have been of no use), B decided to change the operational function to Scout in order to extend their field of vision and collect data ("I go here just to check"). At 15:02 RStr6 was performing Scout by moving west. After 47 seconds of data collection and sensemaking, B concluded that BStr6 could not have moved that far west ("No, it couldn't have") and ordered Rstr6 to instead move east to block any possible attempts from the blue side to bypass.

Discussion

This section summarizes findings and conclusions, and points to directions of future research.

Conclusions

This paper documents an approach to modeling functional interdependency and constraints that addresses the challenges of (1) the adoption of a detailed description of interdependency and associated

understanding of interdependent functions (Brehmer, 2007) and (2) the application of that description to both own and opponent forces' opportunities and vulnerabilities to provide for agility and resilience (Alberts, 2007). The Functional Resonance Analysis Method (FRAM; Hollnagel, 2004) is shown to describe the C² functions of the DOODA loop (Brehmer, 2007) and the tactical and operational functions of military activity. FRAM models have been applied to own and opponent forces in a computer-based dynamic war-game (DKE) to reveal and characterize both agile and unsuccessful C² practice.

More specifically, the paper shows that:

- FRAM's recursive way of functional modeling is suitable for modeling functions at various levels, such as the tactical, operational, and C² (strategic) levels,
- Brehmer's DOODA loop may be developed into detailed specifications of functions through FRAM, enhancing the understanding of military activity and C² functions, and the interdependencies and relations between them,
- FRAM has the potential to describe and analyze functions involved in adversarial C², and enables the analyst to specify the constraints on own and adversary functions, in order to identify strengths and weaknesses in function performance on both sides, which may be used to determine which actions to plan for in order to provide for agile command and control,
- The FRAM methodology has been successfully extended to allow for the description of military activity at the tactical and operational levels and their relationship to command and control functions,
- Data collected during a war-game experimental simulation may be used to develop a functional model, and can be organized in a functional manner following the FRAM function description.

Future work

In this paper, the method has been tested in a war-game microworld environment. The method would need to be further developed, applied, and evaluated in field studies of actual military operations, in order to reach its full potential. Retrospectively, the method may be tested to model and understand past courses of action in order to evaluate which functions and which function aspects were decisive on both sides in the performance of a battle. Prospectively, the method may be tested to model and plan a battle, analyze own and opponent strengths and weaknesses, and evaluate with actual outcomes whether the FRAM function interdependencies descriptions.

Acknowledgments

This work is based on a project funded by the Swedish Defence Materiel Administration (FMV). Professor Berndt Brehmer and the Swedish National Defence College DKE research team, Dr Jan Kuylenstierna, Hans Sandström, and Joacim Rydmark, are thankfully acknowledged for insightful comments, experimental data, and practical support. Professor Erik Hollnagel was instrumental in the development of the ideas presented here. Earlier results were published in a project report, and in the proceedings of the 10th IFAC/IFIP/IFORS/IEA HMS Symposium (Woltjer, Smith, & Hollnagel, 2007).

References

- Alberts, D. S. (2007). Agility, focus, and convergence: The future of command and control. *The International C2 Journal*, 1 (1), 1-30.

- Alberts, D. S., Garstka, J. J., & Stein, F. P. (1999). *Network centric warfare: Developing and leveraging information superiority* (2nd (Revised) ed.). CCRP.
- Brehmer, B. (1992). Dynamic decision making: Human control of complex systems. *Acta Psychologica*, 81 (3), 211-241.
- Brehmer, B. (2005). The dynamic OODA loop: Amalgamating Boyd's OODA loop and the cybernetic approach to command and control. In *Proceedings of the 10th International Command and Control Research and Technology Symposium (ICCRTS)*. McLean, VA: CCRP.
- Brehmer, B. (2006). One loop to rule them all. In *Proceedings of the 11th International Command and Control Research and Technology Symposium (ICCRTS)*. Cambridge, UK: CCRP.
- Brehmer, B. (2007). Understanding the functions of C2 is the key to progress. *The International C2 Journal*, 1 (1), 211-232.
- Brehmer, B., & Allard, R. (1991). Real-time dynamic decision making: Effects of task complexity and feedback delays. In J. Rasmussen, B. Brehmer, & J. Leplat (Eds.), *Distributed decision making: cognitive models for cooperative work*. Chichester: Wiley.
- Brehmer, B., & Sundin, C. (2000). Command and control in network-centric warfare. In C. Sundin & H. Friman (Eds.), *ROLF 2010: The way ahead and the first step* (pp. 45-54). Stockholm: The Swedish National Defence College.
- Brehmer, B., & Sundin, C. (2005). *ROLF 2010: Overall joint command and control in crises and war*. Stockholm: The Swedish National Defence College.
- Cebrowski, A. K., & Garstka, J. J. (1998). Network-centric warfare: Its origin and future. *US Naval Institute Proceedings*, January, 28-35.
- Hollan, J., Hutchins, E., & Kirsch, D. (2000). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction*, 7 (2), 174-196.
- Hollnagel, E. (2004). *Barriers and accident prevention*. Aldershot, UK: Ashgate.
- Hollnagel, E., Nemeth, C. P., & Dekker, S. (2008). *Resilience engineering perspectives, volume 1: Remaining sensitive to the possibility of failure*. Aldershot, UK: Ashgate.
- Hollnagel, E., & Woods, D. D. (1983). Cognitive systems engineering: New wine in new bottles. *International Journal of Man-Machine Studies*, 18, 583-600.
- Hollnagel, E., & Woods, D. D. (2005). *Joint cognitive systems: Foundations of cognitive systems engineering*. Boca Raton, FL: CRC Press/Taylor & Francis.
- Hollnagel, E., & Woods, D. D., & Leveson, N. (2006). *Resilience engineering: Concepts and precepts*. Aldershot, UK: Ashgate.
- Klein, G. A., Ross, K. G., Moon, B. M., Klein, D. E., Hoffman, R. R., & Hollnagel, E. (2003). Macrocognition. *IEEE Intelligent Systems*, 18 (3), 81-85.
- Smith, W., & Dowell, J. (2000). A case study of co-ordinative decision-making in disaster management. *Ergonomics*, 43, 1153-1166.
- Von Clausewitz, C. (1832). *On war*. Project Gutenberg eBook. Available from <http://www.gutenberg.org/> (Project Gutenberg release date February 25, 2006 [eBook 1946]).
- Woltjer, R. (2008). Resilience assessment based on models of functional resonance. In *Proceedings of the 3rd Symposium on Resilience Engineering*. Antibes Juan-Les-Pins, France: MINES ParisTech.
- Woltjer, R. (2009). Functional modeling of constraint management in aviation safety and command and control. Unpublished Doctoral Dissertation, Linköping University, Linköping, Sweden.
- Woltjer, R., Smith, K., & Hollnagel, E. (2007). Functional modeling and constraint management in command and control: two microworld studies. In *Proceedings of the 10th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems*. Seoul, Korea.
- Woltjer, R., Smith, K., & Hollnagel, E. (2008). Representation of spatio-temporal resource constraints in network-based command and control. In J. M. C. Schraagen, L. Militello, T. Ormerod, & R. Lipshitz (Eds.), *Naturalistic decision making and macrocognition* (chap. 17, pp. 351-371). Aldershot, UK: Ashgate.