Application of the Scenario-Based Engineering Process to the Unmanned Ground Vehicle Project *

Erik Mettala
Microelectronics and Computer Technology Corporation
Austin, TX
mettala@mcc.com

Diane J. Cook and Karan Harbison
University of Texas at Arlington
Arlington, TX
{cook,harbison}@cse.uta.edu

Abstract

We describe the Scenario-Based Engineering Process (SEP) that takes a set of scenarios and defines in a formal manner the detailed components required for successful accomplishment of the scenario. This approach can be used to identify technology holes and to define a program that addresses critical research issues. SEP scenarios were used to identify technology needs in the UGV, including planning and control of vehicle formation, sensor planning, and planning for scarce communication bandwidth. We describe the SEP methodology and describes its application to the UGV.

1 Introduction

One of the most difficult responsibilities of a program manager or system developer is identifying technology holes in the program or system before the gaps become chasms of delay and lost revenue. This responsibility then includes evaluating various research programs and technology development efforts and determining their ability to bridge the identified gaps.

Such technology gaps can be found in a majority of program and system development efforts. A successful program is one in which the critical research issues and technology gaps are discovered early in the effort. Discovering holes while the program is underway, or inadvertently selecting non-critical research programs for funding can be expensive in both time and money.

In this document, we describe a more formal and measurable approach to resolving these issues. The approach is called the Scenario-Based Engineering Process (SEP) [3]. SEP can be used to identify technology holes and define a program that addresses critical research issues. SEP can be incorporated along with suggestions from review boards and working groups to identify critical needs and select research projects that will most effectively meet those needs.

The SEP process is used as part of the engineering lifecycle. The approach has been tested on the DARPA Unmanned Ground Vehicle Program and the NCMS/Air Force Mantech/ARPA Next Generation Controller Program, and is currently being expanded to the health care domain. In the next section of this document, we will describe the Scenario-Based Engineering Process in detail. We will then de-

*This research was supported by DARPA contract DAAHO4-93-G-0423.

scribe the application of SEP to the Unmanned Ground Vehicle Program, and will illustrate the process with two RSTA scenarios from the UGV mission.

2 The Scenario-Based Engineering Process

The Scenario-Based Engineering Process focuses on domain-specific scenarios that help transition a problem from the current approach to automation of the approach. SEP focuses on the users throughout the entire systems engineering process. The SEP approach applies to heterogeneous, complex systems, including all the software, hardware, and human components carrying out tasks. SEP can be characterized as being user-centered, architecture-based, iterative, and prototype-focused. The approach provides a systematic transition from the current real world to an envisioned autonomous or semi-autonomous world.

In the case of the UGV, the SEP approach utilizes a set of military scenarios. By formally examining in detail the components of these scenarios, it is possible to derive the technical requirements. Also, working in the reverse direction, one can determine the effect on a scenario of a technical lack of capability of one of the components.

The major activities in SEP include domain analysis, domain-specific architecture definition, application architecture definition, application design and development, and verification and validation. The relationship between these activities is shown in Figure 1. Each of these activities is described in detail below.

2.1 Domain Analysis

An understanding of the task domain is crucial to designing and implementing an effective solution to the problem. Information about the task domain should be used to generate specifications that describe the system behavior unambiguously, consistently, and completely. Various types of domain analyses have been studied for the purpose of generating system requirements. Structured analyses and object-oriented analyses are both popular methods. An additional promising method analyzes scenarios – possible ways in which the desired system will be used to achieve a user's goal.

In the SEP process, scenarios provide the backbone of the domain analysis task. Scenarios are developed to ensure that the important aspects of the domain are understood. In addition to presenting a scenario as a complete thread of execution through a task, scenario components can be represented as constraints describing characteristics of the environment or of the desired application system.

Initially, these scenarios are generated at the highest abstraction level. The scenarios are then refined into smaller scenarios called *tasks*. These tasks contain all the activities that the system must perform within the scenarios. Scenarios must be elicited from one set of experts in the domain, and are validated by an independent set of experts.

2.2 Domain-Specific Architecture Definition

From the domain analysis, SEP produces a domain-specific system architecture. DARPA recently initiated a domain-specific software architecture program (DSSA) with the purpose of developing an open systems architecture-based approach to system engineering. The purpose of DSSA is to increase the efficiency and reliability of new technological developments by providing a structure supporting the reuse of components and domain analysis models. Developing a system by following the path from domain analysis through DSSA to new application areas will result in accelerated advances in new technology.

SEP is an instantiation of the DSSA philosophy. The domain-specific system architecture itself is developed as an activity in this second step of the SEP process. SEP relies on the other activities (domain analysis, application definition, design and development, and validation and veri-

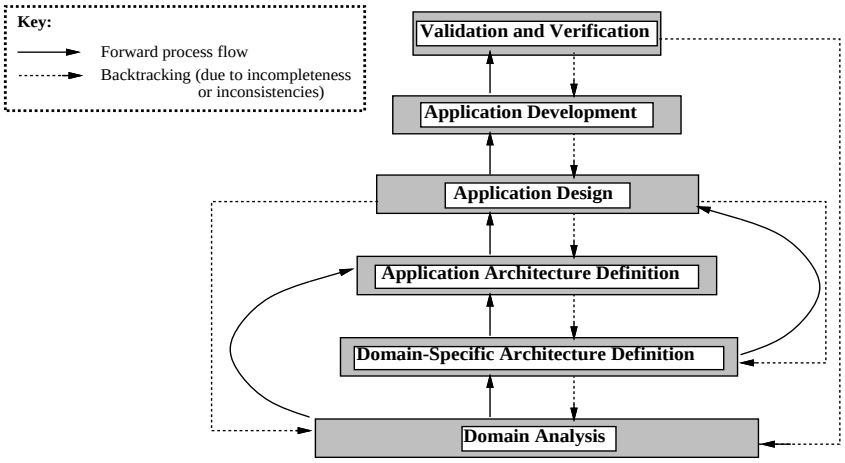

Figure 1: SEP major activities

fication) to support the creation and refinement of the domain-specific system architecture.

The goal of the domain-specific architecture is to specify and abstract the requirements for the tasks and the interfaces between the tasks. The abstraction of such tasks allows a technology solution for a task in one domain to be easily reused for a similar task in a second domain.

This high-level conceptual model specifies both the static and dynamic aspects of the task domain. The static aspects of the domain include the real-world entities and their static relationships and attributes. SEP provides a static analysis of the task domain using object diagrams, where objects represent tasks, actors, and events found in the scenario. The dynamic aspects of a domain are modeled by state transition diagrams, event trace diagrams, and data flow diagrams.

2.2.1 Object Diagrams

An object diagram attempts to model real-world entities in terms of a set of cooperating independent entities, or objects. Each of these objects provides a well-defined set of attributes and behaviors. The scenario object diagram produced by SEP models scenario tasks as objects. In this diagram, objects are represented as boxes.

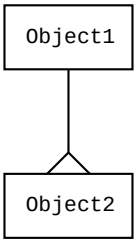

Figure 2: Inheritance

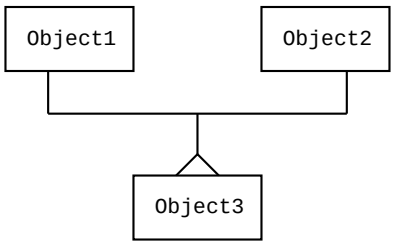

Figure 3: Multiple Inheritance

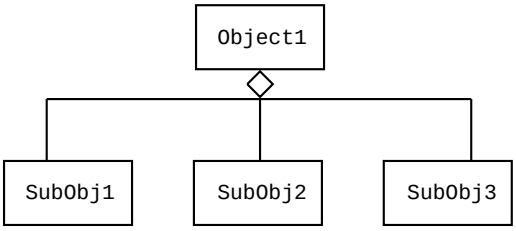

Figure 4: Aggregation

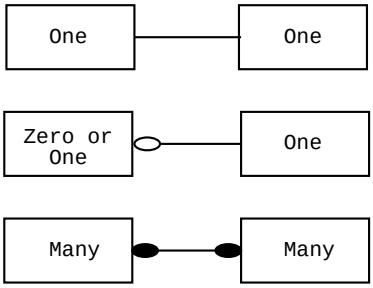

Figure 5: Cardinality

Hierarchies of tasks can be defined using the property of *inheritance*. Figure 2 illustrates the object-oriented representation of inheritance. Through inheritance one task or class of tasks can have access to the same attributes and computational methods as another. Figure 3 illustrates the notation for multiple inheritance, in which a task can inherit attributes and behaviors from multiple parent classes.

Complex tasks can be broken down into simpler subtasks using aggregation. Figure 4 illustrates the object-oriented notation for aggregation. Using this type of analytical tool, a detailed scenario can be broken down into its constituent task parts in order to separate individual functionalities and technology responsibilities.

Figure 5 illustrates the object-oriented notation for cardinality. This tool is used to specify the number of occurrences of a particular task. Each connection may be specified with the exact range of instances allowed. For example, "0:1" allows for zero or one instance, and "0:+" allows for zero or more instances.

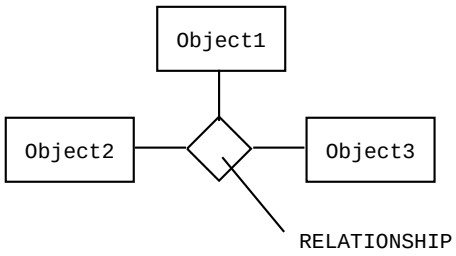

Figure 6: 3-ary Relationship

In addition to the inheritance, aggregation and cardinality types of linkages, objects (representing tasks in this analysis) can be connected to each other to indicate specific relationships between the objects, as shown in Figure 6. The link between objects is usually labeled with the type of relationship that exists between these objects. In some cases, there can exist classes of relationships between objects. A relationship object, which can be part of a relationship class, is represented using a diamond and is linked to all the participants of the relationship.

2.2.2 State Transition Diagrams

A state transition diagram can be generated by SEP to represent the temporal relationships between events in the scenario. The state transition diagram explicitly links tasks that follow each other in the scenario. Multiple tasks within a single state indicate multiple actions to be performed concurrently. The start state for a particular scenario or subscenario is indicated by a black circle, and the final state is indicated by a black circle inside a white circle.

2.2.3 Event Trace Diagrams

An event trace diagram provides a second type of analysis of the dynamic events occurring within the scenario. In addition to representing the sequence of events, explicit communication between tasks and between actors is represented in the event trace diagram. Temporal interactions between objects as well as tasks are indicated in this type of diagram.

2.2.4 Data Flow Diagram

The data flow diagram provides yet another view of the dynamic interactions between objects and tasks in the scenario. Instead of explicitly representing temporal relationships, a data flow diagram represents the information that is passed from one object or task to another throughout the course of the scenario.

2.2.5 Application Architecture Definition

The domain analysis and domain-specific system architecture provide an abstracted view of the system from the users' perspective. The output of these steps can then be used to create an instantiated system or application.

Figure 7 illustrates the transition from domain-specific task scenarios and constraints to a domain-specific system architecture, and from the DSSA to a specific technology application for the task. The process of moving from a domain scenario to a DSSA requires decomposition of the scenarios into tasks and abstraction of the tasks to categories of tasks, as was described in the last few sections. The movement from a domain model (DSSA) to a technology application requires specialization from performance characteristics of the DSSA to performance capabilities of existing or desired technology. The next few sections describe the mapping from a DSSA to the definition, design and development of a technology solution that meets the specified requirements.

The mapping from a general task category to a technology solution is constrained by the performance characteristics of the solutions, as shown in Table 1. Each solution for a task category has strengths and weaknesses and performance envelopes within which it operates. The performance requirements of the task are matched against the performance envelopes of the solutions to determine the solutions with the best fit. General systems performance theory provides a formal means of performing this match. As technology is identified or developed to meet a specific performance criterion, it can be added to a component database for reuse in another domain.

2.4 Application Design and Development

Upon completion of the application architecture definition, the application system is designed and developed. Design requirements, standards, and computational models from the application architecture are used to complete the system. Specific technology components which are used to automate a given task are added to a component database for future reuse.

2.5 Verification and Validation

Verification and validation of the application are performed by independent experts. The performance models specified by the DSSA are used here to perform system validation testing.

2.6 Relationship Between Scenario Development and Technology Development

Note that while SEP flows forward from a domain analysis through the DSSA to application development and verification, SEP relies upon a feedback loop. Feedback from the system validation and verification may be used to specify task requirements initially left out of the scenarios and task description. Similarly, feedback is given to the DSSA based on insights from the application design, and feedback from the application architecture may generate the need for a more detailed domain analysis. Interaction between the domain-specific layers and application-specific layers is also used to identify gaps in current technology. If a specified task cannot be mapped onto existing technology, SEP identifies this area as a critical research need for further funding and development.

This interaction between application layers and domain-specific layers will occur as technology instantiations require information about the task that was not initially considered. In the same way that domain experts supply informa-

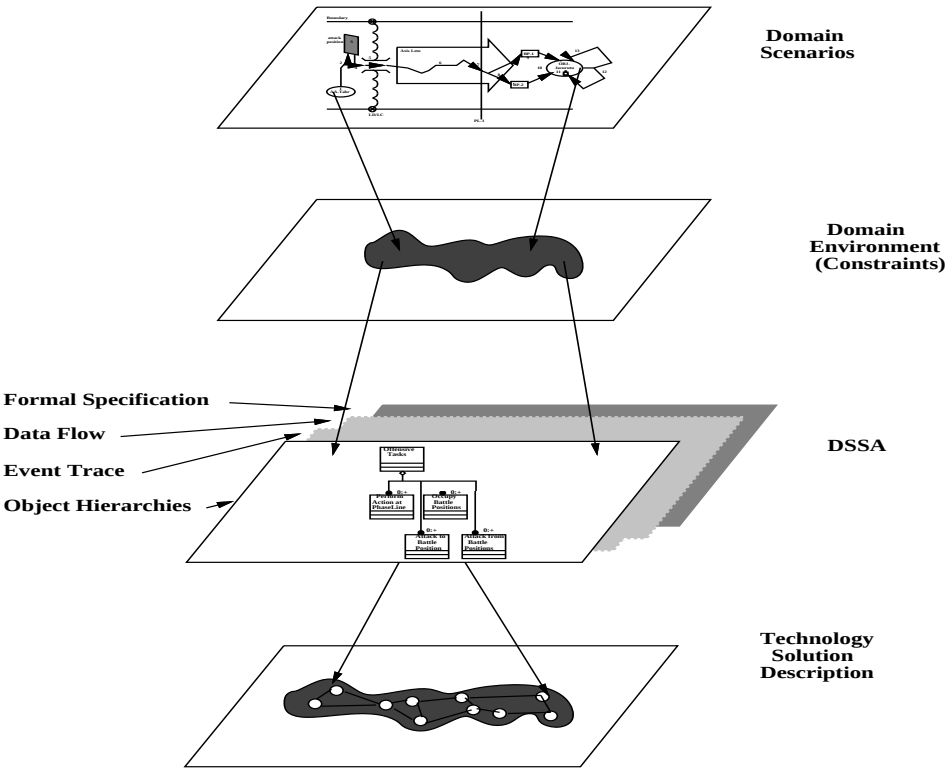

Figure 7: Flow from Domain to Technology

Table 1: Tasks are mapped to technologies and providers

Task Name	**Performance Requirements**	**Technologies**	**Provider**
Stationary target detection	range = 2500 m sensor = 3-5 microns	FLIR	Hughes
	range = 2000 m false alarm rate = 30%	FLIR/Color/Ladar	CSU
Moving target detection using stationary sensor	range = 2000 m false alarm rate = 40%	temporal differencing and stabilization	Sarnoff ARL
Moving target detection using moving sensor	range = 1.5 km false alarm rate = 45%	patchwise stabilization FLIR	Sarnoff ARL
Target recognition	range = 2000 m DB = {M113 APC, M35 Truck, M543 Wrecker, M60 Tank, HMMWV} ID rate = 100%	model-based FLIR	Nichols Loral Vought
	range = 2000m DB = {M113 APC, M35 Truck, M60 Tank}	Ladar	Lincoln Labs
...	...	...	...

tion crucial to the development of a working application, so technology developers cause those experts to focus on detailed aspects of the domain that may be so hard-coded in the expert's approach that they are initially omitted from the domain analysis.

3 SEP Analysis of a UGV Problem

The goal of DARPA's Unmanned Ground Vehicle program is to demonstrate autonomous systems technology including navigation and RSTA (Reconnaissance, Surveillance, and Target Acquisition), and to develop software architectures and reusable technology components for reuse in future unmanned military missions. The culmination of current UGV efforts were realized at the DEMO-II workshop in the spring of 1996.

The natural tendency for large integration programs such as the UGV program is for researchers to use the program as a forum for demonstrating their latest developments. The problem with this tendency is that the focus is on individual accomplishments rather than on the problem at hand, and many large gaps in the program will not be noticed (nor filled) until very late in the schedule. Another problem with this approach is that the resulting system will be defined and described in terms of the researchers' jargon, and the user's needs and understanding will be left out of the loop.

To ensure that the program requirements satisfy the user's needs and identify critical technology areas, we have applied the Scenario-Based Engineering Process to the UGV program. In particular, we have completed the entire process for one offensive and one defensive UGV scenario, and for the concurrent RSTA scenarios.

3.1 SEP for a UGV military offensive task

This section will present a single scenario generated for the Unmanned Ground Vehicle program. A portion of the resulting domain-specific architecture definition will also be presented. In particular, the object diagram, the state transition diagram, the event trace model and the data flow model for a single task within the scenario will be defined.

1. Unit occupies assembly area AA-Tabr in column formation. Unit performs pre-mission planning.
2. Unit moves from AA-Tabr to attack position via traveling (a movement technique) in column formation (a specific formation type).
3. Unit occupies attack position in diamond formation (a specific formation type).
   (a) Unit leader contacts stationary force element commander responsible for passing lane.
4. Unit moves from attack position to LD/LC via traveling (a movement technique) in column formation (a specific formation type).
5. Unit moves across LD/LC while traveling in column formation.
   (a) Unit performs passage of lines with stationary forces.
6. Unit attacks along axis of advance Leto via traveling overwatch in diamond formation.
7. Unit performs actions at phase line PL-1.
   (a) Unit leader observes and confirms that all unit elements have reached PL-1 on time.
   (b) Unit leader reports to command that PL-1 is reached.
   (c) Unless command tells unit leader to hold, unit leader will continue the attack.
8. Two-element teams attach to battle positions BP-1 and BP-2 along axis Leto via bounding overwatch in line formation.
9. Teams occupy battle positions BP-1 and BP-2.

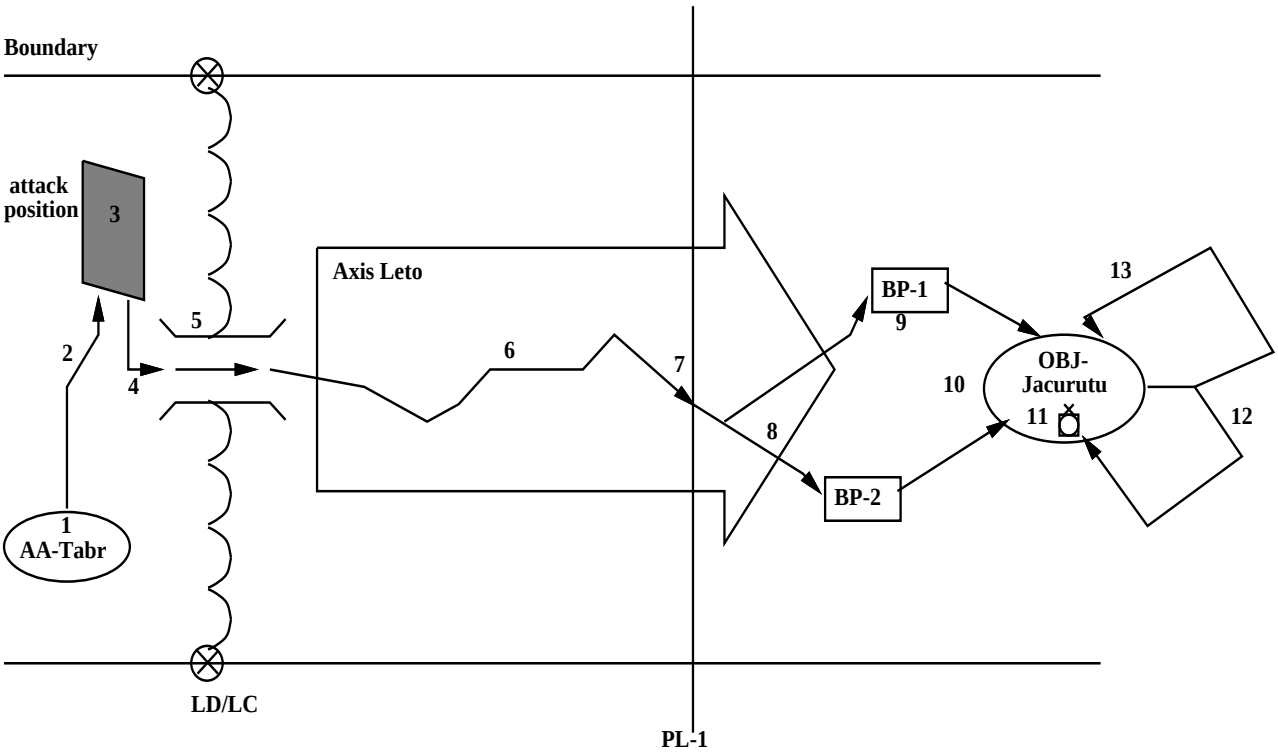

Figure 8: User scenario view

(a) Unit elements perform surveillance of OBJ Jacurutu.

(b) Unit elements perform surveillance of areas left, right, and beyond OBJ Jacurutu.

(c) If enemy forces exist at OBJ, unit elements engage them.

i. Unit element requests indirect fire support.

ii. Unit element adjusts spotting rounds.

iii. Unit elements employ laser target designator for precision delivered weapons.

10. Teams attack from battle positions BP-1 and BP-2 to OBJ Jacurutu via bounding overwatch in line formation.

11. Unit occupies OBJ Jacurutu in diamond formation.

12. Team attacks beyond OBJ Jacurutu via bounding overwatch in line formation.

(a) Team leaders report to command that objective is secure.

(b) If objective is not secure, team leaders report to command necessity to continue the attack.

13. Teams return to OBJ Jacurutu and secure unless otherwise ordered to continue the attack.

(a) In event of extremely successful attack, unit must anticipate immediate commencement of exploitation and pursuit.

Figure 9 illustrates the scenario object diagram for the UGV scenario. This diagram shows that the tasks described in the scenario can be modeled as objects, and that the relationships between these tasks can be specified using object-oriented analysis techniques.

Let us consider a particular task described in the UGV scenario. Figure 10 illustrates Step 9

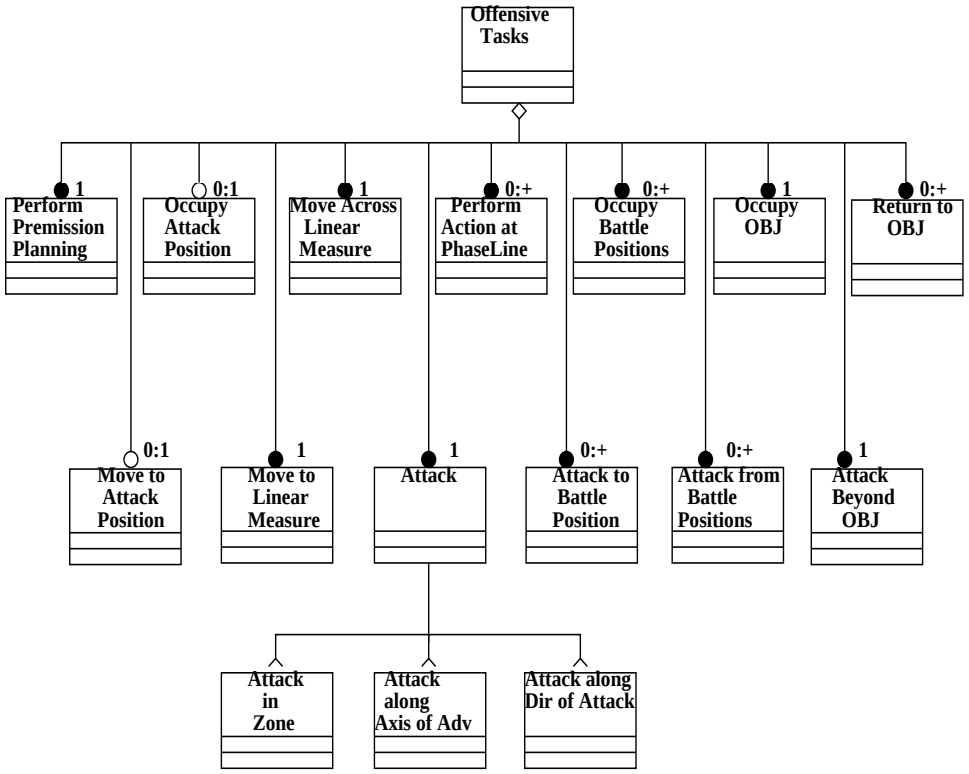

Figure 9: Scenario object diagram

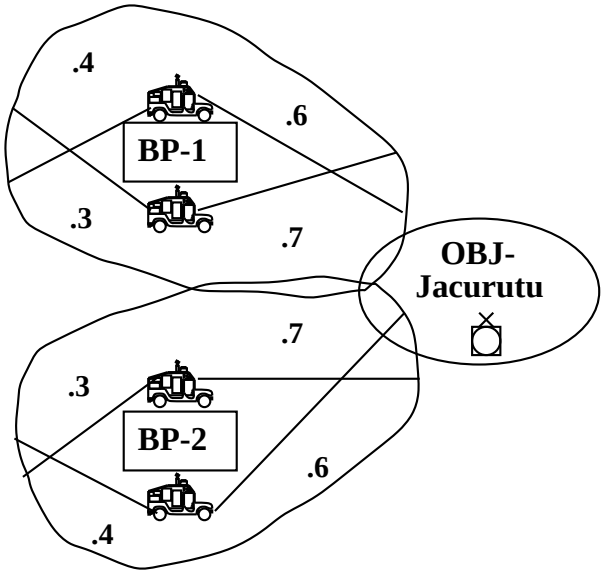

Figure 10: Scenario task

in the scenario "teams occupy battle positions BP1 and BP2", the task on which we will focus. In providing the scenario, the user indicated an interest in describing actions of the UGVs at battle positions overwatching an objective. These actions consist of first observing the area within the objective "Jacurutu", then conducting a search of the areas to the left, right, and beyond the objective. As Figure 10 illustrates, weights are allocated to each vehicle's field of regard. These weights indicate the proportion of time spent in the corresponding field of regard in order to support the user's requirement for 360 degree security. If enemy forces are spotted at the objective, unit elements will conduct an indirect fire task, which may require the use of a laser target designator.

This particular scenario task will now be analyzed using an object diagram, a state transition diagram, an event trace diagram, and a data flow diagram. The object diagram in Figure 11 shows that in this task a unit is an aggregate of two or more teams, that each team

is composed of two or more vehicles, and that each vehicle has an aggregation of RSTA sensors, each of which control an area of observation. In this task of the scenario, the unit is performing surveillance of the objective area, and may send a request for indirect fire support.

In the object diagram, an area of observation is specified in terms of the number of fields of regard, the set of angles defining each field of regard, and the percentage of time spent on each field of regard by the sensor. These descriptors are sufficient to calculate where a specific type of RSTA sensor should be pointing at a specific time in response to a specific mission plan. The information pertinent to these tasks is provided by the RSTA planner, which derives its constraints from the mission planner.

The SEP analysis of this task reveals that the RSTA planner is responsible for coordinating sensor movements and scheduling activity for the unit. Input to the coordination process includes actuator speed, the unit formation, the fields of regard for a vehicle in a specific formation, and the percentages of time allocated to each field of regard.

Each team separately occupies a battle position, so the object model associates each team to an area measure through the link "occupy". In turn, the "area measure" object denotes that a team occupies an battle position whose boundaries are specified as a geometric polygon and whose identifier is supplied by the attribute "name".

Figure 12 provides a state transition diagram representing the sequence of events within Task 9. The left-hand side of the figure describes the mission state transition, and the right-hand side of the figure describes two simultaneous RSTA activities. While the mission events specify surveillance of the areas within and around the objective, the RSTA events describe first an "area of observation" corresponding to the execution of search control between fields of view within a set of fields of regard, and then a "RSTA sensor" corresponding to the continuous movement of a sensor to a specified direction and the capture of an image at the appropriate time.

Figure 13 illustrates an event trace model for Task 9. Each column heading represents a significant interacting object in the context of the task. Communications between objects are represented as arrows between the corresponding object columns. In this analysis, events are ordered temporally from the top to the bottom of the diagram.

Figure 14 presents a top-level data flow diagram for the actions the user has specified once the unit occupies the desired battle position. As the diagram illustrates, the position of the objective must be supplied to the sensor planners. When targets are identified, the targets are recorded in the target database, and a target engagement task is evoked. If no targets are identified, each vehicle examines its assigned field of regard, and the task is complete.

3.2 Interactions Between UGV Scenario Development and Technology Development

As shown in Figure 1, user-specified scenarios can be used to pinpoint the need for a given technology. Conversely, the current state of the art in technology may impact the type of scenario that a user can realistically generate. By integrating both scenario development and technology development into a feedback loop, users and technology providers can converge on a feasible and effective system design.

This relationship between scenarios and technology can be witnessed in the SEP analysis of the UGV program. The scenario definitions identified technology needs that were not being met. Similarly, currently available technology defined the types of tasks that could reasonably be expected in the scenario.

Some of the technology needs that were identified by the UGV scenarios included planning and control of UGV formations, sensor planning, and planning for use of scarce communi-

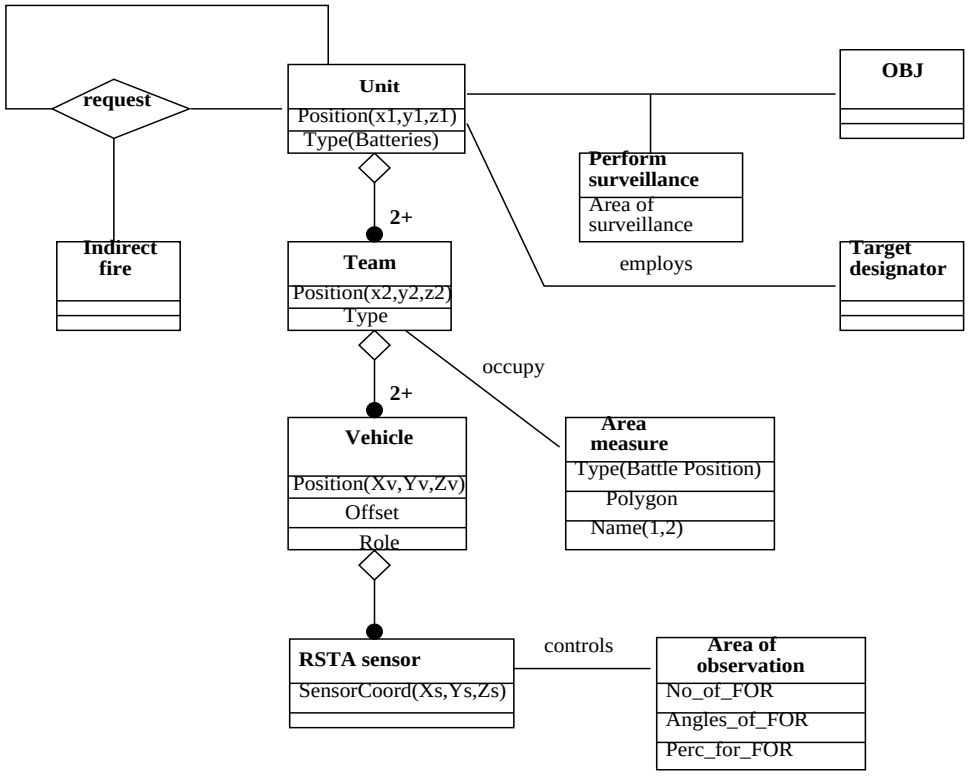

Figure 11: Object diagram for scenario task

cation bandwidth. Early in the program, researchers in the planning and the RSTA groups developed ideas independently from the other group. As a result, the technologies needed to merge these two areas were missing. After the SEP analysis identified these technology holes, specific researchers were tasked with meeting these needs.

In the same way, current RSTA hardware and software capabilities were used to develop realistic UGV scenarios. In particular, the UGV scenarios specify a fixed rate at which the camera will move from one field of view to the next. This rate was determined by the rate at which frames could be grabbed and processed. In addition, the amount of specified overlap between fields of view (a one degree overlap) is dependent on the granularity of control that the camera's rotational actuator will provide. Specifications for these individual RSTA tasks are continuously revised as the available technology is improved.

4 Conclusions

In this paper we have introduced the Scenario-Based Engineering Process and demonstrated its application to the Unmanned Ground Vehicle program. Application of SEP to the UGV program, the Next Generation Controller program [1], and the health care program [2] has revealed that development of an effective program relies on user specification of scenarios that drive the requirements specification, the application development, and the validation and verification of the system. SEP can also be used by a DARPA program manager to define a research and development program. Currently, no formal guidance is available to the manager; he works from only the somewhat informal recommendations of committees and working groups.

Scenario-based engineering lends strength to program development by maintaining user involvement throughout the entire system life cycle. SEP also benefits program development by identifying technology gaps that appear when

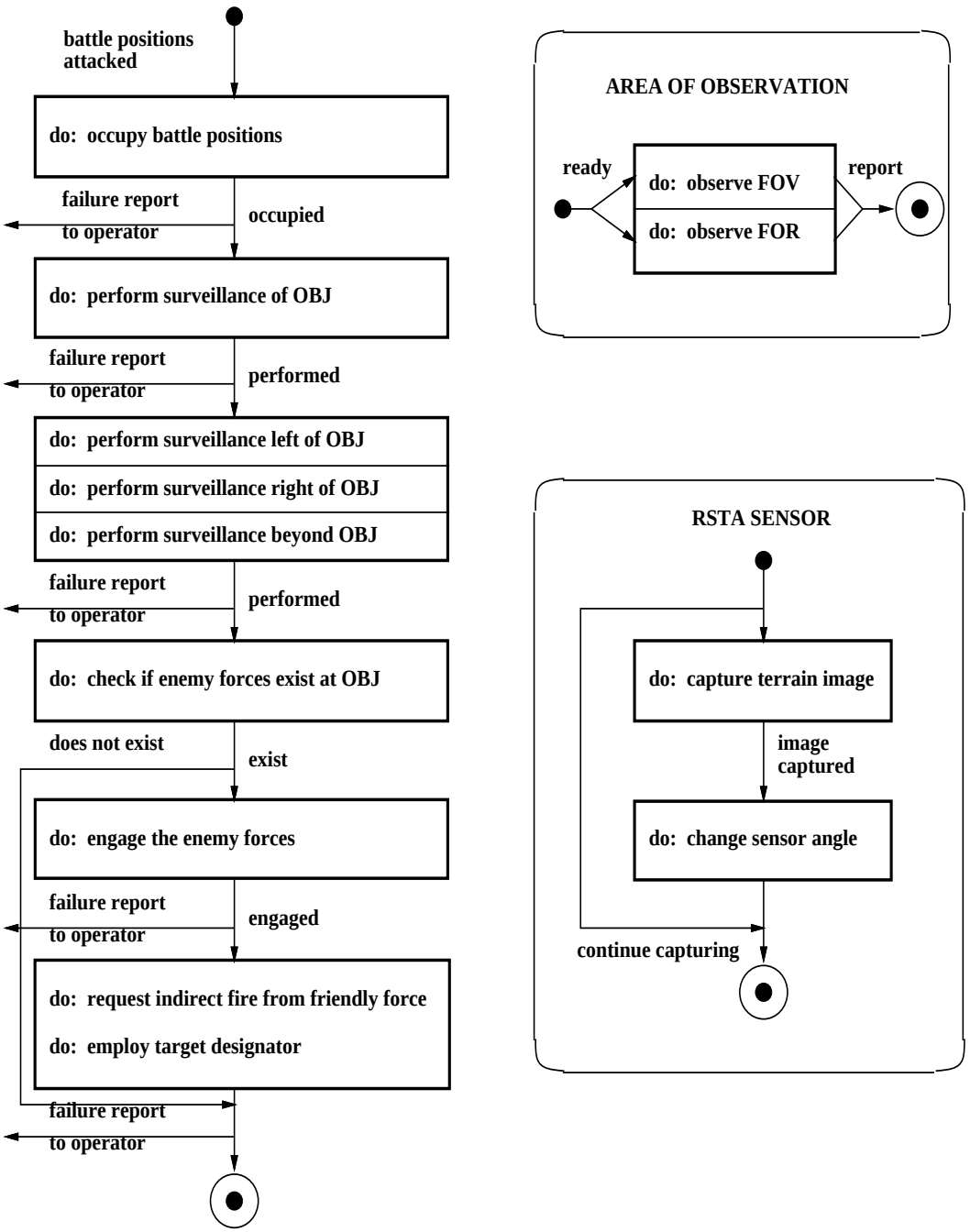

Figure 12: State transition diagram for scenario task

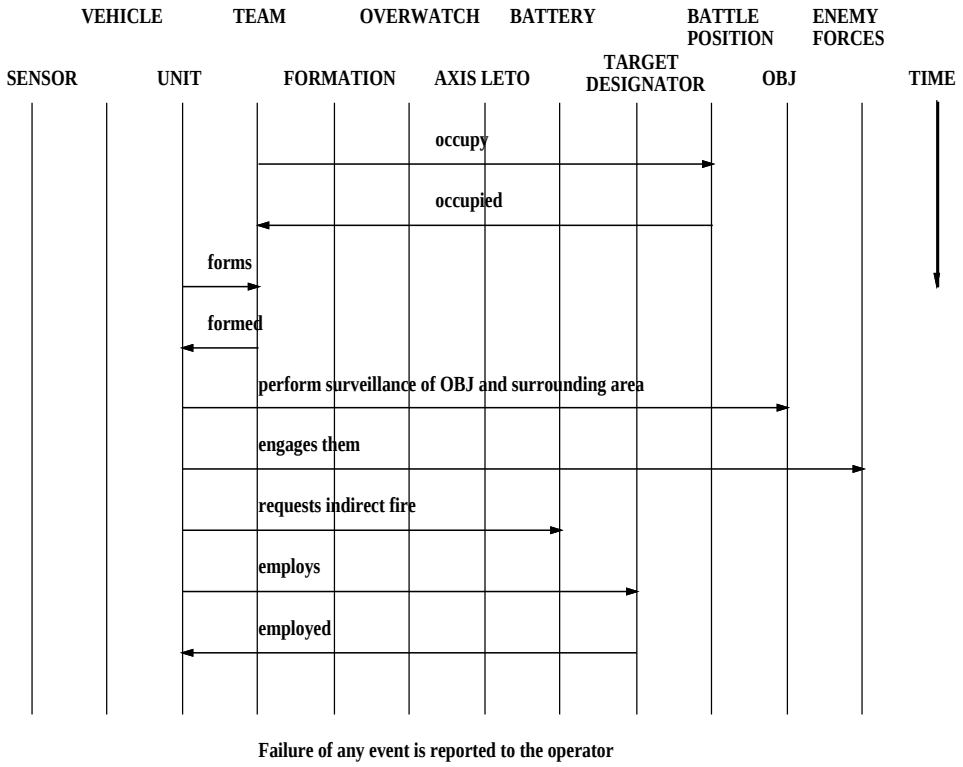

Figure 13: Event trace diagram for scenario task

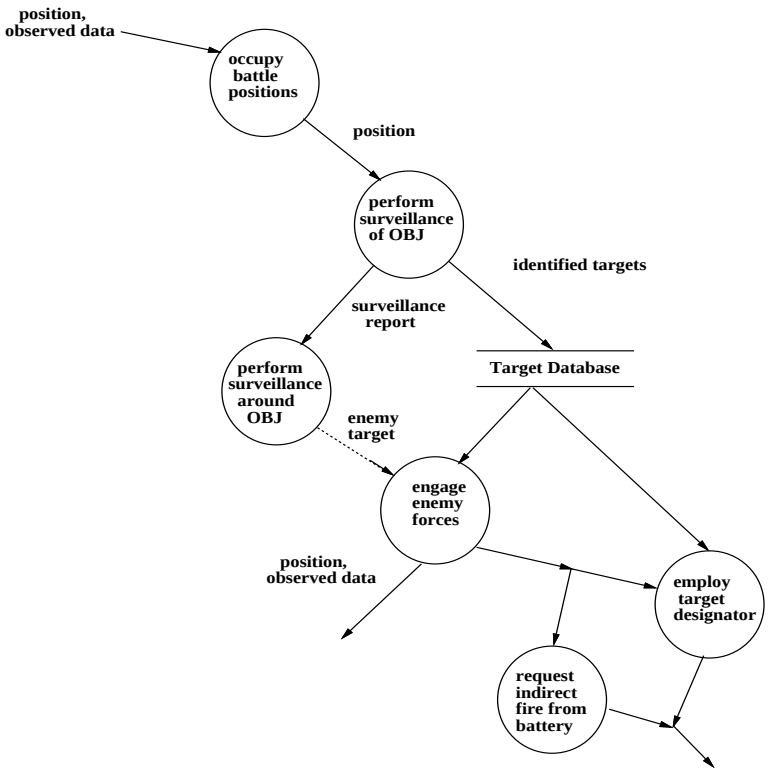

Figure 14: Data flow diagram for scenario task

mapping scenario capability requirements to technology solutions. In particular, we feel that application of SEP to RSTA tasks within the UGV program can identify the critical technologies and integrations necessary to realize a successful DEMO-II and future missions.

References

[1] K. S. Barber, O. R. Mitchell, and K. Harbison. An object-based system incorporating representations and reasoning mechanisms to plan manufacturing applications. *Journal of Systems Engineering*, 5:36–47, 1994.

[2] K. Harbison, L. Burnell, J. Kelly, and J. Silva. The scenario-based engineering processor (sep): A user-centered approach for the development of health care systems. In *Proceedings of MedInfo95*, 1995.

[3] K. McGraw and K. Harbison. *Knowledge Acquisition using the Scenario-based Engineering Process.* Lawrence Erlbaum Publishers, New York, 1995.