

The Common Prototyping Environment: A Framework for Technology Integration, Evaluation and Transition

Mark H. Burstein, Richard Schantz
BBN Systems and Technologies

Marie A. Bienkowski, Marie E. desJardins
SRI International

Steven Smith
CMU Robotics Institute

Abstract

The ARPA/Rome Laboratory Planning and Scheduling Initiative was organized, in part, around a Common Prototyping Environment (CPE), established to promote experiments involving the integration of planning-related software systems. It included a repository containing contributed software available for sharing and reuse, domain information and sample problems, knowledge representation tools, user-interface tools, distributed knowledge-based communications tools, and tools for metering experiments to evaluate system performance. Following a series of preliminary Technology Integration Experiments (TIEs), the CPE Testbed was developed. It is a platform on which to experimentally integrate planning and scheduling systems, in preparation for their transition to operational prototypes. It supports experiments in which compatible planning tools can communicate across the Internet in a shared knowledge representation "interlingua" (KRSL). It facilitates the collection and evaluation of metrics on individual and combined system behaviors.*

Introduction

The ARPA and Rome Laboratory sponsored Planning Initiative (ARPI), is a government funded research program that began in 1990. Its long range objective is to develop large-scale distributed, collaborative, automated and semi-automated military planning and scheduling systems. The Planning Initiative was, from the outset, organized in part around the notion of a **Common Prototyping Environment (CPE)**. The CPE was designed to:

- give ARPI's researchers and developers access to a common body of support software, development tools, and sharable domain knowledge and data, thereby fostering the development of reusable planning and scheduling technologies.

- support experiments that integrate these technologies to solve realistic planning problems; and
- ease the transition of these technologies into operational prototypes through demonstrations of interoperability.

BBN Systems and Technologies and ISX Corporation were teamed to provide the CPE, supporting the research community by providing access to realistic military planning problems, while reducing each individual project's overhead by providing a suite of sharable software tools that would foster the integration, evaluation and transitioning of the planning and scheduling technologies developed. The **CPE Testbed**, the primary focus of this article, was developed in 1993-4, as a distributed environment for continuing the series of **Technology Integration Experiments (TIEs)** begun in previously. The CPE Testbed, consists of a suite of software tools that enable systems implementing specific technological techniques to be experimentally integrated in a distributed knowledge-based communications environment.

The CPE development team also supported the community through the **CPE Repository**, the first element of the CPE to be developed. The CPE Repository is an "electronic clearinghouse" for software (sources, executables, and sometimes both) developed under ARPI auspices, and representative data and knowledge, in both textual and electronic forms, about the military transportation planning processes. BBN had primary responsibility for the Testbed and some of the software used in integration experiments and demonstrations, while ISX has primary responsibility for the Repository.

The CPE Testbed grew out of experiences in collaborative projects from the first several years of the Planning Initiative. These included our participation in the cooperative development of software demonstrations called **Integrated Feasibility Demonstrations (IFDs)** (see accompanying article) and smaller-scale, **Technology Integration Experiments (TIEs)** where different research and development teams explored the issues involved in successfully combining their research software and techniques.

The first set of TIEs occurred during 1992. They addressed a range of related transportation planning and

* (c) Copyright 1995 IEEE. This is an edited version of an article in *IEEE Expert*, Vol. 10, No. 1, pp 17-26, February 1995. Reprinted with permission from IEEE.

scheduling problems grounded in the scenario from the just completed IFD-2 (see accompanying article). One additional TIE, called the "Infrastructure TIE" focused on applying the techniques for database and knowledge-base access and for distributed, knowledge-based systems intercommunications (KQML) being developed under ARPA sponsorship. For that TIE, BBN and ISX worked with Unisys Federal Systems (now a part of Loral Corp.) to develop the initial version of a software infrastructure that became the core of the CPE Testbed, including the first prototype of a "knowledge server".

The CPE Testbed was first released in May of 1993. It capitalized on all of the TIEs conducted during 1992, bringing the software used in all of those TIEs into a common distributed environment and using a single knowledge representation language for information exchange (KRSL).

The CPE Testbed was also designed to support experiments in which components were combined in novel ways, or directly compared doing similar functions. To address the issue of experimentation, the testbed included a suite of tools for collecting and analyzing data about the performance of the modules used in solving planning problems. The CPE Testbed includes technology components developed in more than ten different R&D laboratories sponsored under the initiative. These software systems all provide services that can be utilized in experiments based on solving military planning problems, even when the software systems are physically hosted at sites dispersed throughout the country.

The CPE Repository

From the outset, ARPI promoted the sharing of data and software development tools among the Initiative's research and development contractors, both to better focus on a common set of domain-specific problems, and to reduce the amount of effort put into redundant "tool building". The CPE Repository was established to serve as a central clearinghouse for these tools and information. Housed on a UNIX workstation system and accessed by FTP, it was the primary means of collecting and disseminating data, reusable software development tools, demonstration prototype software systems, and software collected or developed by the initiative's contractors.

The CPE Repository was established in order to help coordinate the many software development efforts to take place at different laboratories throughout the country. In anticipation of the issues that arise when software systems developed in different places are to be integrated, some community "standards" were established for basic hardware and software underpinnings of the prototype software to be developed, covering such things as the type of UNIX workstations to be used, the version of Common Lisp to be used, and the tools used for interface development (the Common Lisp Interface Manager). Other standards were set for such things as X-windows, TCP/IP, etc.

On top of these basic necessities for software development, a range of software tools were added to the repository over time by BBN/ISX and other participating Planning Initiative contractors. These included such things as:

- graphical interface tools (knowledge-base network browsers, a scientific graphics plotting package, a map display system, supplied by BBN),
- system development tools (a standard LISP DEFSYSTEM, logical pathnames),
- a knowledge-base maintenance system (**LOOM** from USC/Information Sciences Institute),
- a knowledge-directed database-access mechanism compatible with LOOM (**LIM/IDI** from Unisys),
- a knowledge-based systems intercommunications protocol (**KQML** from Unisys), and
- an object-oriented distributed communications mechanism that works with many different hardware platforms and programming languages (**CRONUS** from BBN)

Using these and other tools from their own laboratories, as appropriate to their research agendas, a number of other "technology packages" were developed by initiative contractors and installed in the CPE Repository together with documentation. Some contributions were primarily technology demonstrations, while others were intended for use directly in other development efforts.

Among the contributions, there were one or more implementations of

- Generative planners (**SIPE** from SRI, **O-Plan** from the University of Edinburgh),
- Temporal information maintenance systems (**TMM** from Honeywell, **Tachyon** from GE Center for Research and Development),
- Constraint-based scheduling systems (**OPIS** from CMU Robotics Institute), and intelligent database-access and database query planning systems (**SIMS** from USC/Information Sciences Institute, **CoBase** from UCLA).
- Decision support and analysis tools (**DEMOS** and **DT** from Rockwell International's AI Laboratory),
- A knowledge-based systems development tool for reasoning with uncertainty (**PRIMO** from GE CRD) and
- A statistics and metering package to support quantitative experiment analysis for AI systems (**CLIP/CLASP** from the University of Massachusetts).

Over the course of the initiative, a number of military planning applications were developed using these largely "domain-independent" AI tools that tailored or adapted them to the military transportation planning domain. Table 2 briefly describes the ones that were later incorporated into the CPE Testbed. Many of these systems were developed for or in conjunction with IFDs or TIEs.

As a clearinghouse for domain knowledge and data about the military's transportation planning process, the repository also included a large body of textual materials detailing the processes and sample problems related to joint transportation planning derived from military officer's

training courses on the subject. It also included data files containing a variety of sample datasets, taken primarily from unclassified military training materials and IFD demonstration scenarios.

KRSL and The Shared Domain Ontology

ARPI also sought to promote a consistent way of accessing all of the data and different kinds of representations of objects, actions, time relations, and plans, that are inputs or products of the military planning process. As a result, during the project's second year, many ARPI members jointly developed the **Knowledge Representation Specification Language**, (KRSL, pronounced just like 'carousel'), as a kind of "interlingua" that could be used to describe the shared domain data in the repository. In addition to basic object and concept descriptions, KRSL includes forms for units of measure, general relations and propositions, temporal relations, plan/goal descriptions, and producer-consumer constraints.

On top of this basic definitional syntax, substantial fragments of a uniform **Shared Domain Ontology (SDO)** were developed, describing all of the data elements and object types that participated as inputs or outputs to the systems that could be coupled in solving sample planning problems. This ontology is a vocabulary of objects and relationships spanning all of the different kinds of transportation resources, ports, equipment, and other data structures that participate in the planning process. It originally handled all of the information involved in the kinds of planning problems captured during the IFD-2 demonstration. It was later extended with additional concepts necessary to describe noncombatant evacuation operations (NEO) plans and air campaign plans.

As mentioned earlier, the CPE repository includes a large volume of data in its original textual and fixed-format ASCII forms. To make the data more uniformly available, a concerted effort was made to translate selected subsets of that data into KRSL using the concepts in the SDO, so that researchers and developers could draw on a consistently described reservoir of domain data, and appeal to a consistent vocabulary for that data. This standardization process began during the TIEs, and was fully realized in the CPE testbed. In the CPE Testbed, all communications between modules were TCP/IP based messages in terms of the KRSL language, and using the ontology's vocabulary.

Technology Integration Experiments

In the spring of 1992, following IFD-2, planning began for a series of Technology Integration Experiments (TIEs) that would develop and demonstrate interoperability between emerging initiative technologies, collect information about trade-offs between technical alternatives, and validate various technologies for potential use in IFDs. The initial set of TIEs demonstrated most of the elements of infrastructure and planning technology to be introduced into the Common Prototyping Environment over the succeeding 24 months.

The TIEs were explicitly experimental; they explored methods of combining technical components that play specific roles in planning or problem solving, and test designers' claims for their technology's applicability. When possible, we sought to contrast different mechanisms providing the same or similar functions, such as maintaining and reasoning with temporal information. For example, in one tie, two systems for temporal constraint maintenance, TMM and Tachyon, were made to interoperate with a plan generation system (SOCAP) during course of action generation. By sharing a standard interface, the two could be directly compared performing the same task. Other TIEs focused on exploring the how other pairs of complementary technologies could mutually benefit from interoperation.

Each TIE included a 'Final Exam' in which the performance of the technical components involved was to be measured against some baseline criteria. These Final Exams was intended, in part, to help identify component technologies for insertion into an IFD, and, in part, to benchmark performance on typical military planning and scheduling problems. Table 1 lists the TIEs that were completed during 1992-3. The first four contributed to the development of the initial CPE Testbed.

Figure 1 places these ties in the context of a functional "waterfall" model of the overall military plan development process. At the top of the figure is TIE #1, which was a first attempt to prototype several key infrastructure components; knowledge-directed information access and a knowledge-representation based distributed communications infrastructure. This TIE laid the groundwork for all of the components participating in the planning process to be able to rely on a uniform means of accessing and communicating information about the plans under construction. It introduced the notion of a "knowledge service", an intermediary between planning systems and data and knowledge bases.

The three major functional areas of plan development activity are also represented in the figure. Course of action generation (COAG) is the first, deployment plan expansion the second, and plan analysis the third. COAG involves strategic and tactical planning for both employment and deployment of forces. In the TIEs, SOCAP, the SIPE-based plan generator did most of the planning for COAG. Different TIEs examined how the *quality of plans* produced by SOCAP could be improved by integrating SOCAP/SIPE with three different kinds of external functionality: temporal constraint management, case-based reasoning, and constraint-based resource analysis. In a sense, the use of temporal reasoners and constraint-based schedulers in this regard was an attempt to get early feedback to the COAG process from different kinds of automated *plan analysis tools*, the third functional area.

The second major functional area in the TIEs was plan/force expansion, the process of producing detailed deployment plans from high level inputs about major force groups to be deployed (the output of the COAG process). TIE #3 used constraint-based scheduling techniques to develop reasonable deadlines and choices for mode of

transport (air, sea) for the forces during deployment planning.

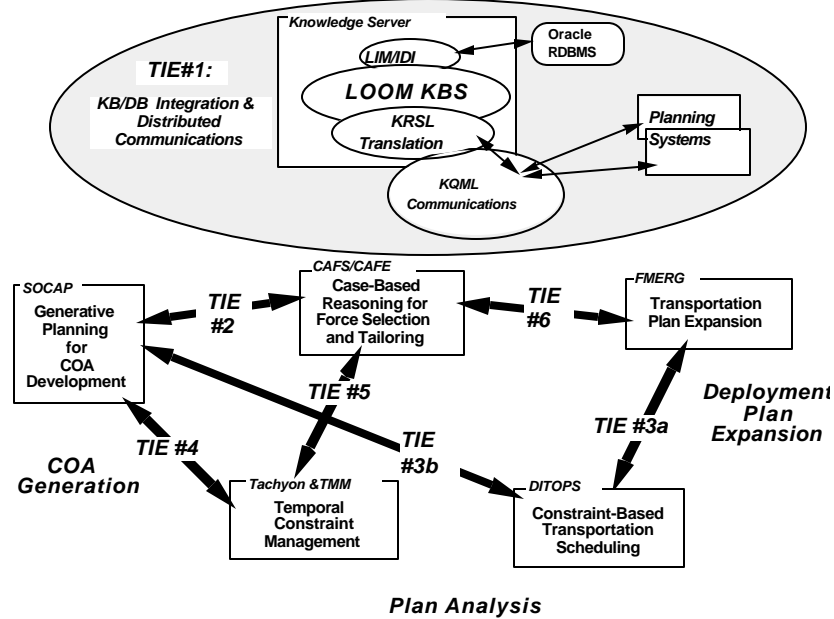


Figure 1: Relationships between TIEs in a functional architecture.

1. **Infrastructure TIE (BBN, ISX, Unisys)** demonstrated the knowledge services and distributed, knowledge-based communications elements of the envisioned distributed CPE planning environment.
2. **Case Based Reasoning - Generative Planning TIE (GE-CRD, SRI)** applied a CBR system developed at GE to select force units for roles in plan operators of military operations plans built by a generative planner.
3. **Constraint Based Scheduling - Deployment Planning TIE (CMU, BBN, SRI)** demonstrated constraint-based scheduling system in a heterogeneous military planning system. **TIE #3A** used deployment constraints to do the final stages of deployment plan development. **TIE #3B** used the scheduler for preliminary deployment plan analysis during the initial COA development, as a means of filtering the options based on transportation resource constraints.
4. **Temporal Reasoning - Generative Planning TIE (GE-CRD, Honeywell, SRI)** used a temporal reasoning system (Tachyon or TMM) during generative planning to propagate the temporal constraints in a developing plan. This arrangement enabled some comparative experiments with Tachyon and TMM.
5. **Temporal Reasoning - Case-based Force Expansion (GE-CRD, BBN)** applied temporal constraints during case-based force expansion, to ensure that units that would be available at the time they were needed.
6. **Case-based reasoning - Force Expansion (GE-CRD, BBN, MITRE)** used CBR techniques to refine and enumerate the elements of forces selected during high-level planning.

Table 1: The Initial TIEs

- **SOCAP (SRI)** An application of the SIPE-2 generative planning system to military employment planning, SOCAP consisted primarily of a domain model and operators for a range of military plans similar to the IFD-2 scenario.
- **CAFS (GE CRD)** An application of GE's Case-based Reasoning tool, CARET, that would select major force units for an employment plan. CARET uses GE's PRIMO system for uncertain reasoning.
- **DITOPS (CMU Robotics Institute)** A re-engineered version of the OPIS constraint-based scheduling system, geared to military deployment transportation scheduling and transportation resource capacity analysis.
- **KTS (Kestrel Institute)** A very efficient constraint-based scheduling algorithm for transportation scheduling, developed using Kestrel's KIDS semi-automatic program generation environment. Contrasted in experiments with DITOPS.
- **PFE (BBN and CMU)** A LISP implementation of a military transportation simulator developed for ARPI, modeled after the ones used by the military to estimate the feasibility of a "TPFDD" deployment plan, given a set of available transportation resources. PFE's data input and output requirements were used as a model for both DITOPS and KTS.
- **FMERG (BBN and ISX)** FMERG was developed for IFD-2 to expand SOCAP-generated plans into military "TPFDDs" or deployment plans. It is a simplified model of a force package elaboration and augmentation system, consisting of subsystems for expanding major force units, and a rule-based system for adding combat support and sustainment elements.
- **FORMAT (MITRE)** A knowledge acquisition tool for retroactive annotation of Force Modules, to enable their indexing and reuse, as by a case-based reasoning system (or a person). - added during CPE Release 2

Table 2: Prototype Military Planning Tools incorporated into the CPE Testbed

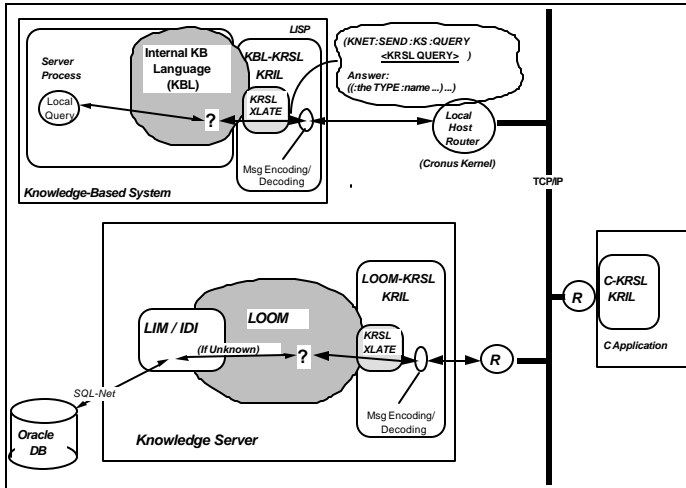


Figure 2: CPE Distributed Communications Model

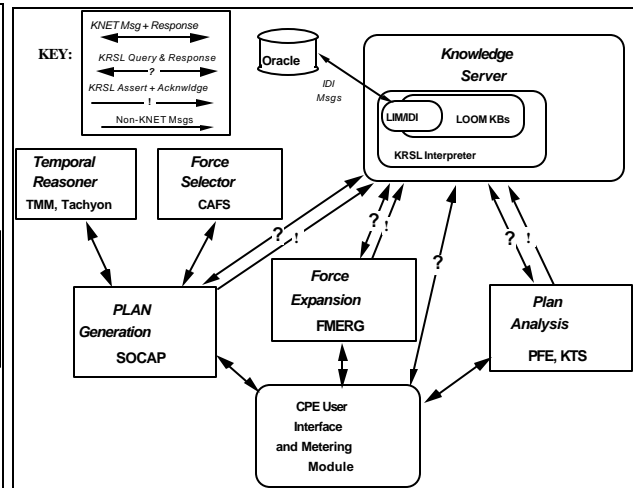


Figure 3: KRSL Communications paths in CPE Testbed Release 1.0

The Infrastructure TIE: Distributed Knowledge Access

The goal of the Infrastructure TIE was to demonstrate integration and interoperability technology for the Common Prototyping Environment in particular and for ARPI software in general. It was developed under the premise that knowledge-based software systems could interoperate most effectively with other software systems in a distributed environment using a consistent representation language or 'interlingua'. They could thereby avoid having to develop and maintain independent means of translating messages between every pair of communicating systems. The TIE demonstrated the feasibility of this integrated approach to inter-agent communications and the sharing of knowledge among knowledge-based software agents and knowledge-level access to common databases. It formed the basis for the first working prototype of the CPE infrastructure.

ISX developed an implementation of the KRSL language, the 'KRSL Kernel' to act as this 'interlingua'. The KRSL Kernel translated forms, queries and assertions into and out of the LOOM knowledge representation system, developed at USC/Information Sciences Institute (MacGregor and Yen, 1989; MacGregor and Burstein, 1991). To construct a 'knowledge server' to play the role of information broker in the distributed architecture, LOOM was also integrated with Unisys' Intelligent Database Interface (IDI) through an intermediate layer called LIM (LOOM Interface Module). This provided access to a set of persistent relational databases maintained in Oracle's (TM) RDBMS.

Figure 2 shows the general architecture of the knowledge server and communications substrates developed during the Infrastructure TIE and used later in the CPE. The TIE communications software was built around Unisys' implementation of KQML, a distributed communications model for intelligent software-agents developed from a specification that was the work of a number of AI

researchers under the auspices of the DARPA Knowledge Sharing Project. KQML supported the message encapsulation and routing of messages composed of KRSL expressions. During the TIE, messages were passed among the Knowledge Server, FMERG and PFE, the modules that had participated in the previous IFD-2 demonstration.

Communications between these agents, consisting primarily of representations of military plans and deployment schedules were standardized as much as possible using what then existed of an evolving Shared Domain Ontology (SDO), a common set of terms, concepts and relations defined in KRSL to be used for all communications between agents in the CPE environment. This catalog of definitions included standard frames or concepts for such things as air ports, sea ports, geographic locations (called GEOLOCs), transportation movement (TPFDD) plan elements, and specific representations required to enable communication among the modules involved, e.g., the major force list representation of SOCAP plans and the representation of force modules developed from IFD-2.

The Infrastructure TIE also developed the first implementation of a Knowledge Server (KS), an agent whose primary purpose was to act as a maintainer of shared knowledge and information, and a mediator to databases containing additional information. The Knowledge Server consisted of the KRSL Kernel, LOOM, the Intelligent Database Interface (IDI), and the LIM interface between LOOM and IDI. IDI also communicated with an external Oracle data base containing most of the relevant data from IFD-2, including geolocs, force descriptions, etc. The Knowledge Server supported external queries and assertions in the KRSL interlingua, communicated via KQML. The information in the answers it provided could either come directly from the LOOM knowledge base or from the Oracle data base, via LIM and IDI.

The TIE recreated a portion of the IFD-2 demonstration, this time using KRSL and KQML to access and store intermediate information and results in the Knowledge Server. A SOCAP major force plan was asserted into the knowledge server using KRSL. FMERG was invoked, and issued KRSL queries to the knowledge server for the major force plan and definitions of the forces involved, and received that information back in KRSL forms based on the SDO definitions of the objects involved. It then produced a TPFDD for that plan, and asserted that result back into the Knowledge Server, from which it could be retrieved by PFE for a transportation feasibility analysis. This Infrastructure model allowed components to reside on the same workstation or be distributed across the Internet. For example, the demonstration used FMERG running at Rome Laboratory, communicating over the Internet with a Knowledge Server running at Unisys in Philadelphia.

Much of the initial infrastructure was instrumented to collect timing information. As might be expected for a first prototype, some steps were quite slow, especially given the size of the data that was transmitted (e.g., 20 minutes to retrieve approximately 1MB of information about forces for a plan). This information served as a guide to us in the subsequent development of the CPE Testbed.

The integration of KRSL, LIM and LOOM also raised some representational issues and inconsistencies that needed to be addressed. One of the more interesting of these was the issue of how to handle the relationship between class-level information as stored in database records and the corresponding classes or concepts in the knowledge base. For example, a database table might contain information about types of aircraft (e.g., their average speed, cargo capacity, etc.) which were best represented as attributes of class-level concepts in LOOM or KRSL although LIM could only map database records onto instances at that time. Also, KRSL objects with represented subparts could not be derived properly from information in the data base, because of the well known difficulty relational databases have with cyclic or recursive relations.

Using External Reasoning Tools in support of Generative Planning

A single, uniform-strategy AI reasoning tool, it is not always sufficient to tackle a real-world problem. In some cases, legacy software must be integrated with the AI system; or the AI system must be altered to fit the new problem (e.g., with a customized user interface); or additional automated reasoning techniques must be added to round out the capabilities of the AI system for work in that domain. Three TIEs supplemented a mature generative planning tool with several other substantial pieces of AI technology: a case-based reasoner; a temporal reasoner, and a scheduling system. The goal was to improve the quality of the plans generated by SOCAP.

TIE #2 used a case-based reasoning (CBR) system, CAFS (Case-based Force Selection), to extend SOCAP's ability to choose objects (force units) to participate in operations.

CAFS selected appropriate forces from a library based on similarity comparisons between those forces and the operator requirements and context provided by the new plan. TIE #3B integrated SOCAP's planning capabilities with DITOPS (Distributed Transportation Scheduling in OPIS), a constraint-based scheduling system developed at Carnegie Mellon University. SOCAP's ability to generate robust, feasible plans with realistic allocation of resources was improved by using feedback from the scheduling system directly during planning. TIE #4 experimentally combined two temporal constraint propagation systems (Tachyon, TMM) with SOCAP to improve SOCAP's temporal reasoning capabilities. This integration resulted in SOCAP being able to represent more sophisticated temporal constraints within plans and to reason more accurately about the times and durations of actions and about resource utilization over time.

These integration projects were unique in two ways: first, they utilized existing, independently developed AI-based modules to supplement an existing generative planning system; second, they add capabilities that are novel or relatively unexplored in generative planning systems.

Temporal Reasoning in support of Generative Planning with SIPE-2

When SOCAP was originally developed as an application of SIPE-2, SIPE-2's limited temporal reasoning capability surfaced as a shortcoming. Before the TIE, SIPE-2 was unable to reason about utilization of resources and could not place temporal constraints between actions in the plans. Consequently, the plans generated did not represent important constraints that existed in the planning application domain.

In previous versions of SIPE-2, time was treated strictly as a consumable resource; that is, it could be consumed but not produced, and its consumption over parallel tasks was nonadditive. Each action specification could have associated start-time and duration variables, but SIPE-2 calculated specific values for time variables only when the constraints forced a particular value; otherwise, the allowable range was computed.

SIPE-2 had several techniques for establishing the relative orderings of actions: inserting ordering links to avoid resource conflicts; using one action to meet several other actions' precondition requirements by ordering those actions after the action that achieved those conditions; and coordinating separate subplans by adding ordering links between subgoals of the two subplans. These capabilities allow many simple temporal problems to be solved, but the inability to represent the time constraints of two possibly unordered actions was seen as a problem. Two SIPE-2 actions are either ordered with respect to each other, or they are unordered. If the latter, the planner considers it possible to order them either way or to execute them simultaneously. It was not possible to model *when* the various effects of an action become true during its execution, or that must occur *simultaneously*. By adding as a support system a temporal constraint manager for Allen's

(Allen, 1981) temporal relations calculus, it became possible to explicitly represent actions starting or finishing at the same time, overlapping each other, or one occurring during another. Many dependencies between different military actions should be represented in this way in SOCAP. For instance, cargo off-load teams should arrive at the same time as the first air or sea transport arrives at the destination airport or seaport.

For the TIE, SOCAP's ability to represent and reason about time was extended by adding a layer on top of SIPE-2 that would keep track of the temporal constraints within the plan, initially using the Tachyon temporal reasoning system (Arthur, Deitsch and Stillman, 1993) to maintain and propagate these constraints. The interface to Tachyon was designed to be general enough to permit a different temporal reasoner to be substituted (in particular, TMM (Dean & McDermott, 1988; Schrag et al. 1992).

Our approach to the integration of a temporal reasoning engine was somewhat "coarse-grained". A new plan critic was written for SIPE-2 to be run at the end of each "planning level", rather than as each new constraint was encountered during planning. This critic extracts all of the temporal information (time windows and inter-node constraints) from the plan, sends it to a Temporal Reasoner, and stores the updated time windows returned back into the nodes in the plan. In addition, methods were added to maintain these constraints on the plan as goals are expanded to a new level by SOCAP. SIPE-2's operator syntax was extended to allow a designer to specify any of the 13 Allen relations or quantitative constraints (the permissible range of metric distances) between the endpoints of any pair of nodes in an operator.

The extended system found temporal inconsistencies in previously generated plans which could only be resolved by changing the dates on which military units were available to perform missions, or by assigning different units to those missions. This shows that the system now reasons with a more complete model of the constraints operating in the military domain. The temporal information is especially important for proper use of the planner's products by downstream scheduling software, since SOCAP is able to pass a more complete and consistent set of constraints to the scheduler. Although the temporal reasoning introduced into SIPE-2 by the TIE cannot be considered "complete" since representation and reasoning support for both continuous and interruptible events are still lacking, the limited capabilities that were added provide a significant source of power in dealing with important constraints in the domain of military planning and improve the generated plans substantially.

Case-Based Reasoning for Force Selection

Selecting the right force to participate in a military operation and tailoring a force to meet special requirements of a specific operation are two important parts of operations planning. Considerations such as a unit's potential to deter or defend against an enemy threat, its mobilization, its ability to handle the terrain, and its time to deploy must be

reviewed. When it was initially developed, SOCAP always asked the user to select the force units that would be used to achieve specific missions, by presenting a list of available units that met the constraints of the operator (to perform the mission) when expanding a goal for which that operator applied. The user could see what constraints were met by the units in the list but had to rely on personal knowledge of the mission context, and capabilities of the each possible unit to determine which was most appropriate in that context. SOCAP itself had no way of ranking or preferring some units over others if they all satisfied the *necessary* constraints on their utilization for a given job.

This kind of preferential force selection and tailoring was seen as an area where case-based reasoning (Kolodner, 1993) would apply. TIE #2 was an addressed this issue by integrating a CBR system from GE's Center for Research and Development called CAFS (CAse-based FEselection) which was built on GE's CAse-based REasoning Tool (CARET). In CAFS, descriptions of forces from a case library of force units are indexed and retrieved based on their mission requirements, climate, terrain, mobility and other related information.

For TIE #2, SOCAP was modified to call the CAFS module for major force selection instead of presenting a list to the user for selection. CAFS uses the constraints and context information provided by the SOCAP operator and bindings to find and rank potential force units as to their appropriateness for a given mission (including ones that were tailored to specific missions in the past). If the closest matching force does not fit SOCAP's requirements, heuristics can be applied within CAFS to modify the force in the appropriate manner. For example, it might be necessary to add additional support units to the retrieved force structure to account for differences between the prior and current situation.

Constraint-based Scheduling as a Capacity Analysis Tool

An important feature of the type of decision support that SOCAP is intended to provide is a capability for replanning based on feedback from (usually external) plan feasibility evaluation tools. Assessing the transportation feasibility (essentially the use of resources over time) of a plan is a major point of focus for military planners. To investigate this issue, and also explore ways of overcoming SOCAP's simplified model of resource management, TIE #3 explored an integration of SOCAP with CMU's DITOPS system for constraint-based resource reasoning and scheduling. In this TIE (#3B), SOCAP was modified to call DITOPS at various stages of its search through the space of possible plans to assess the feasibility of the current partial plan from the standpoint of transportation resource capacity requirements. This early analysis aids in the assignment of resources to operations based on projections of resource bottlenecks. That is, either SOCAP or a user can use the analysis results to choose feasible deployment destinations for major forces during initial plan generation or to reassign transportation resources.

To focus DITOPS' attention on the relevant parts of the plan, SOCAP extracts a temporally ordered plan network that included only the transport operations and other non-resource-using actions that contributed temporal constraint information. For each operation, it supplies the transportation resources (planes, ships) assigned to transport specific units, and holding capacity required on those resources to transport the personnel and materials involved. This information is analyzed by DITOPS' capacity analysis routines and the result is passed back to SOCAP in the form of comparative estimates for the expected availability versus demand for aggregate resources, by class of transport, over time. This data is presented on a color graph showing expected demand on a resource type vs. available capacity over time. Users (and, eventually, SOCAP itself) can then reassign resources or change the time of an operation.

Force Elaboration and Transportation Scheduling

A TPFDD (Time-Phased Force Deployment Database) is the current-day operational data form of a military deployment plan. It is a table of all of the individual units to be deployed by air or sea to specific destinations, their sizes (in tons of air or sea cargo), and the dates that they are available for shipment and need to arrive at their destinations. TPFDDs are used as inputs to simulators to estimate the transportation feasibility of a plan in plan analysis tools such as DART. They are also input to scheduling algorithms used to produce detailed schedules and manifests for shipments of air and sea cargo. They are thus a critical link in the data path for current military planning operations.

The current-day process of developing a TPFDD is a labor-intensive function, primarily performed by the command staffs of individual services (Air Force, Army, Navy). Once a general operations plan has been developed by the Joint Staff, the services are tasked to "fill in the details" of the operations, and determine the lists personnel and equipment that will be needed. They often do this by of "cut and paste" operations on detailed force lists developed for prior missions, although this is largely a paper and pencil and "fat finger" data entry task without the proper software support tools. Since TPFDDs can consist of thousands or tens of thousands of records, this is a labor-intensive task, indeed.

In order to fill out the schedule for the deployment of units in a TPFDD, service planners estimate the travel times and itineraries of each represented unit. Frequently, they do this with only approximate information about the transportation resources that will be available, and even less information about the constraints on loading and unloading of cargo at individual ports.

They are forced to do this because they have no way of *specifying directly what they do know*, namely the ordering and timing constraints (e.g., which units need to arrive before other units) that are based on their knowledge of the plan under construction and military planning more

generally. In the Planning Initiative, we sought to demonstrate a path to automate some of these staff functions, by showing how automated tools could help, given the right information at the right time. In IFD-2, BBN and ISX developed FMERG as a very simplified first step toward automated TPFDD generation directly from operations plans produced by SOCAP. FMERG approximated parts of this currently manual process, expanding each force mentioned in the SOCAP plan into a list of units and materials by using a force module library, augmenting the list of operational units to be deployed with a number of other units serving combat support and service support functions (local transportation, medical, food services, etc.), and, finally, adding units to the deployment plan based on doctrinal rules about resupply for things like food, fuel and ammunition (this last activity is more fully realized in the system LOGGEN, developed by MITRE).

Two TIEs were designed as next steps in this development process. In one, GE-CRD extended their Case-Based Reasoning tool, CAFS, to provide more capability to do intelligent tailoring of those configured units to adapt them to new situations. In the other TIE, CMU developed scheduled departure and arrival time information for TPFDDs using the DITOPS constraint-based transportation scheduling tool.

Constraint-based TPFDD Scheduling

DITOPS is a constraint-based scheduling system for military transportation planning based on the OPIS system for job-shop scheduling (Smith, 1994). In the job-shop scheduling domain, there are typically a number of ways in which manufacturing of an item might be accomplished, by using sequences of operations, each of which can be accomplished by using tools that exist at multiple stations on the factory floor. In transferring this model to transportation scheduling, DITOPS was made to work with largely completed TPFDDs, which specified the goods to move, and required destinations, but could leave open some of the decisions about *which resources* would be used to move the goods, and, potentially, *which intermediate ports* the goods would travel through. DITOPS then built detailed schedules of which ships and planes will be used to transport each good, in effect combining a transportation feasibility analysis similar in spirit to what military simulators had done before with the development of actual schedules.

TIE#3 opening up the possibility that DITOPS would take over some of the decision making relating to port and mode (air or sea-based transport) assignment, by removing the overly restrictive prescriptions embodied in a TPFDD. These kinds of choices are the analog in the transportation planning problem of the choice in the job-shop world about which of several paths through the factory can be used to complete a manufacturing task in the most timely fashion.

For this experiment, DITOPS used a less constrained description of the plan than that normally embodied in a TPFDD, leaving choices to be made about which ports ships could be destinations for each load, and which kinds

of transportation resources could be used (air or sea craft). Normally, the military uses a rule of thumb that, for large operations, 20 percent of the cargo should move by air, and 80 percent by sea. However, given timing considerations and available lift options, this heuristic can easily be wrong. By relaxing some of these constraints on the shape of the scheduler's output, but providing more information about the structure and dependencies in the plan, more efficient plans and schedules can often be produced, and they are certainly more readily revised.

DITOPS was provided with some additional information about the doctrinal constraints on which elements of a force needed to move before other elements, replacing fixed, but estimated, dates for unit movements found in a TPFDD. Given this additional information, DITOPS was able to demonstrate the automating of mode and port decisions based on resource capacity information. DITOPS was able to balance the use of ports, and avoid some bottlenecks due to port capacity overload. Similarly, by avoiding early commitments on how cargo was to be shipped (air or sea modes) the scheduler was able to find efficient means of transportation for each unit (for example, using excess air transport capacity to improve arrival times) and better balance the limited resources available. This TIE successfully demonstrated how a more appropriate encoding of the available information could lead to better, and more easily and automatically revisable, transportation plans.

The CPE Testbed

For the first IFDs and TIEs, ARPI contractors developed and used largely domain-independent AI tools to address military planning problems. After the initial round of TIEs was completed, we began to bring together all of the technology components involved into the CPE Testbed, a single environment based on the interoperability model used in the Infrastructure TIE. The CPE Testbed was designed to:

- Provide an infrastructure for future TIEs that supported experimental metrics and their evaluation and uniform access to data and scenarios for the transportation planning domain.
- Establish a common representational basis for all remote communications between planning and scheduling systems.
- Facilitate the transition of technologies into IFDs and operational prototypes.
- Enable the exploration of coordination and control issues in a heterogeneous system for replanning and rescheduling.

We developed CPE Testbed Release 1.0 in an intense five month effort during the spring of 1993 involving all of the participants in TIEs. During that time, all of the ad-hoc communications that has occurred in the TIEs were transformed into consistent KRSL representations, including the development of KRSL plan representations and a translator from SIPE-2 plan representations to KRSL forms. We developed support for remote starting and

metering of software modules, in part by replacing some of the underpinnings of the early prototype KQML software with CRONUS, a more mature object-oriented communications substrate that supported many languages, and by adding a simple, KQML-like interface called KNET.

Substantial effort and voluminous EMAIL traffic was devoted to standardizing the language of communications between the modules using KRSL. To standardize the functionalities of the different systems for experimental comparison purposes, all communications between modules were organized around standard sets of messages to be defined for pairs of functional classes of software modules. Each system was classified as one of six types of modules: Plan Generators, Force Selector, Temporal Reasoner, Force Expander, Deployment Scheduler (or Simulator) or Knowledge Server. For each type of module, we defined the KNET messages that it would have a contract to respond to, in KRSL syntax. Figure 2 describes the modules that were incorporated into the CPE Testbed. Figure 3 shows the interaction paths along which communications between module types were defined.

Our attempt to standardize the content of communications between modules was largely successful, although time constraints forced some hard choices on the process. In particular, the communications between plan generators and the temporal reasoners continued to have the batch-oriented flavor of the original TIE interface between SOCAP and Tachyon, even though TMM, the other temporal reasoner, was designed for more incremental interactions with clients. Similarly, there were some aspects of SIPE's plan language that are unique to that system which carried over into KRSL and the SDO.

Other improvements over the system represented by the Infrastructure TIE included a better message encoding scheme that achieved a three-fold reduction in the size of large messages, and a ten-fold decrease in transmission time of those messages. This meant that the communication of a TPFDD object of transmission size over 1.5 MBytes during the TIE was reduced to .5Mbytes and the time reduced from 20 minutes to 2 minutes. Additional speedups occurred in accessing remote databases from the Knowledge Server as well.

The Testbed as a platform for technology evaluation.

The CPE Testbed was designed to run experiments that tested the speed and effectiveness of new technologies, both singly and in TIE-like combinations. To run an experiment in the CPE, one first chooses sets of modules that will run in each of a number of 'trials'. One defines the data to be collected using a mechanism for adding 'alligator clips' or **meters** at various points in the computations of modules, and in their communications with other modules.

By simple declarations, information about run times and process parameters can be collected on a trial-by-trial basis, where each trial might use a different module to performing the same problem solving function. For example, by setting up the sequence of trials in the CPE Testbed Experimenter's

User Interface, Tachyon and TMM were compared in sets of trials where they were called by SOCAP with the same sequence of inputs. To evaluate the results of experiments, the interface incorporates the CLASP analytical statistics package from the University of Massachusetts. Thus, data collected by CPE meters can be directly analyzed using a variety of statistical measures.

Experiments can also use the CPE User Interface to specify the hosts on which modules will run during a trial. All active hosts in the same CPE configuration, be they at BBN in Cambridge, ISX in Los Angeles, Rome Laboratory in New York state, or one of the developer sites at other locations around the country could be used during a single trial, if the modules involved were installed at those sites.

At the time this article was first written, there were twelve modules different "modules" or "agents" that can communicate via the CPE's distributed communications model. These systems represented the contributions of ten different corporations and university laboratories, and several others will be added shortly. In addition to the "CPE User Interface" module, they were: SOCAP, Tachyon, TMM, CAFS, DITOPS, KTS, FMERG, PFE, FORMAT, the Knowledge Server, and an experimental "Plan Server" for plans developed by the TARGET plan authoring system from IFD-3.

Conclusion and Current Status

The CPE, including both its testbed and repository aspects, successfully demonstrated the potential for increased collaboration and sharing among a large set of research and development projects. It served as a validation of the concept and methodology for promoting more effective software and information sharing, moved a large community of researchers toward mechanisms for validating the products of their work both in terms of increased effort to demonstrate interoperability and increased attention to the establishment of experimental metrics for progress.

The CPE is no longer under development for Phase 3 of the Planning initiative. Some in the university research community found the CPE to be less directly useful to them than we had hoped, partly because of when it became available, late in their research cycle, partly because of its unwieldy size, and partly because of the compromises we made to support existing, rather than next generation, software systems. Most of the systems that were incorporated into the testbed were the applications of AI technologies that pre-dated the Planning Initiative. We were less successful in addressing the widely varying needs of the university researchers developing the next generation of tools.

Devising an architecture for a diverse, heterogeneous population of software systems all participating in solving complex planning problems is a challenging task. It is in some ways even more challenging when one understands that there will always be a strong human element to these distributed systems. The challenge we now face is not just to build knowledge-intensive distributed software systems, but systems in which many people, at many different

locations are assisted by intelligent planning software in their dynamic, ongoing efforts to coordinate the activities of one of the largest single organizations in the world.

Acknowledgments. The portion of this work performed by BBN and ISX was funded by ARPA and monitored by Rome Laboratory under contract F30602-91-C-014. We would also like to thank our colleagues at Carnegie Mellon University, General Electric, Honeywell, Information Sciences Institute, Kestrel Institute, MITRE, SRI, and Unisys (now Loral), for their contributions to the work described.

References

- Allen, J.F. "Maintaining knowledge about temporal intervals." *Communications of the ACM*, 26(11):832-843, 1983.
- Arthur, R., Deitsch, A., and Stillman, J. "Tachyon: A constraint-based temporal reasoning model and its implementation." *SIGART Bulletin*, 4:3, July 1993.
- Bienkowski, M.A., Desimone, R.V., and desJardins, M.E., "Decision Support for Transportation Planning in Joint COA Development: Annual Report." SRI International Technical Report ITAD-2062-AR-93-101. March 1993.
- Dean, T.L. and McDermott, D.V., "Temporal Data Base Management, *Artificial Intelligence* 36:1-55. 1988.
- MacGregor, R. and Yen, J. *The Knowledge Representation Project at ISI*. ISI Research Reports ISI/RS-88-199, July 1989, University of Southern California.
- MacGregor, R. and Burstein, M.H., Using a Description Classifier to Enhance Knowledge Representation. *IEEE Expert*, 6:3, June 1991, pp. 41-46.
- Wilkins, D.E., *Practical Planning: Extending the Classical AI Planning Paradigm*. Morgan Kaufmann Publishers, Inc., San Mateo, CA. August 1988.
- Wilkins, D.E., and Desimone R.V., "Applying an AI Planner to Military Operations Planning." In M. Fox and M. Zweben, eds., *Intelligent Scheduling*. Morgan Kaufmann Publishers, Inc., San Mateo, CA. 1993.
- Schrag, B., Carciofini, J., and Boddy, M. " β -TMM Manual (version b19)." Technical Report CS-R92-012, Honeywell Systems & Research Center, 1992.
- Allen J. F. et al. *Reasoning About Plans* Morgan Kaufmann Publishers, Inc., San Mateo, CA. 1991.
- Kolodner, J. *Case-Based Reasoning*. Morgan Kaufmann Publishers, Inc., San Mateo, CA. 1993.
- Smith, S.F., "OPIS: A Methodology and Architecture for Reactive Scheduling", in (Eds.) Fox, M. and Zweben, M. *Intelligent Scheduling*, Morgan Kaufman, Palo Alto, 1994.