AF2008098

Abstraction Techniques for Capturing and Comparing Trajectory Predictor Capabilities and Requirements

Robert A. Vivona[*]
L-3 Communications, Billerica, MA 01821

Karen T. Cate[†] and Steven M. Green[‡]
NASA Ames Research Center, Moffett Field, CA 94035

Recent research has increased focus on the conceptual design, development and use of air- and ground-based aircraft trajectory prediction capabilities to support advanced Air Traffic Management concepts. In both the United States and Europe, the sharing of four-dimensional trajectory information between many automation systems will be necessary for successful operations. Understanding the functional and performance differences between disparate trajectory predictors is critical for enabling this system interoperability. Documented capabilities for four existing trajectory predictors were compared to identify commonalities and differences. For effective comparison, it was first necessary to abstract the prediction capabilities of each trajectory predictor. Three abstraction techniques were developed. The first separated the description of modeled aircraft behavior from the associated mathematical models used to integrate the predicted trajectory. The second defined a conceptual boundary between the trajectory predictor and its client application. The third eliminated the use of domain specific terminology. The abstraction techniques proved not only beneficial for comparing trajectory prediction capabilities, but also for defining trade-offs between the compatibility and accuracy of disparate TPs to achieve system interoperability.

I. Introduction

Recently, the global Air Traffic Management (ATM) research community has increased its focus on the conceptual design, development and use of aircraft trajectory prediction capabilities, both on the ground and in the air, to support advanced ATM concepts. In the United States, the Next Generation Air Transportation System (NextGen) identifies “Aircraft Trajectory-Based Operations” and “Performance-Based Operations and Services” as two of the eight capabilities needed to achieve its goals of increasing the safety, security, and capacity of air transportation operations.[1] Trajectory-Based Operations utilize predicted aircraft four-dimensional trajectories to “assess the effects of proposed trajectories and resource allocation plans, allowing both service providers and operators to understand the implications of demand and identify where constraints need further mitigation.” Performance-Based Operations and Services provides a foundational change in the National Airspace System, defining regulations and procedural requirements in performance terms rather than in terms of specific technology or equipment, “thus potentially maximizing the value of service providers’ and users’ investment” in existing equipment. Similarly in Europe, the Single European Sky Air Traffic Management Research (SESAR) identifies the four-dimensional trajectory as the core of their future ATM system, with the main driving principle of “each single flight [being] executed as close as possible to the intention of its owner.”[2] In both the NextGen and SESAR concepts, the sharing of trajectory information between many ATM automation systems will be necessary for successful operations.

Understanding the functional and performance differences between disparate trajectory predictors (TPs) is critical for enabling the interoperability between these systems. If trajectory and aircraft intent data are to be

[*] Chief Research Engineer, Associate Fellow AIAA.
[†] Research Scientist, Senior Member AIAA.
[‡] Research Scientist, Associate Fellow AIAA.

effectively shared,, the variation in requirements, including input data (e.g., intent) and output data (trajectories), must be properly understood. This understanding is necessary to determine the minimum data that must be shared to achieve interoperability without degrading performance. Also, by evaluating the commonality between the TP requirements and capabilities of different automation systems, it may be possible to enhance this commonality through the elimination of unnecessary differences. None of this is possible without first understanding what are the range of requirements and capabilities for a representative set of current TPs.

The ability to cross-compare TP requirements and capabilities over a range of ATM concepts and decision support tool (DST) automation of interest will enable the identification of their commonality. To this end, a team of companies with expertise in TP technology (Lockheed-Martin, L-3 Communications, and GE-Aviation) documented the capabilities of five TPs from different domains, environments and functional support (see Table 1).[3,4,5,6,7] The five TPs were chosen to cover a range of current day TP applications, including air- and ground-based support of control, traffic flow management (TFM) and flight planning systems in both research lab and operational deployment environments. These documents all followed a common document structure, referred to as the Requirements Framework, which was developed by the National Aeronautics and Space Administration (NASA), Lockheed Martin and L-3 Communications to enable a clear, consistent, complete and cross-comparable documentation of TP requirements and capabilities. The ultimate goal is to develop a mature version of the Requirements Framework through a phased process of documenting and comparing a range of TP technologies, of which these first five documents represent the initial set. While a formal comparison of this first set of documents is not scheduled until after the publication of this paper, an informal initial comparison of several key aspects of four of the documents (see Table 1) was conducted. The lessons learned by performing this initial comparison provided significant insight into the analysis, documentation and comparison of TP capabilities and requirements.

The purpose of this paper is to present the lessons learned and insights gained about how to describe TP functional requirements and capabilities. In particular, several techniques developed to abstract the description of TP capabilities, enabling more effective comparisons of disparate TP capabilities, are described. In addition, insight gained into use of these abstraction techniques to solve a wider range of TP issues, such as identifying critical data for effective system interoperability, is also discussed. The paper begins with a discussion of the background work that motivated these efforts, followed by an overview of the lessons learned that led to the development of the abstraction techniques. The main contribution of the paper is presented next through detailed descriptions of the three main abstraction techniques developed. For each abstraction technique, the benefits to maturing the Requirements Framework as well as benefits beyond that effort are presented. Finally, some additional, broader benefits and issues uncovered while analyzing TP capabilities using abstraction techniques are described.

TP	Domain	Environment	Functionality	Included in Initial Comparison
CTAS En Route[3]	Ground	Research	Air Traffic Control, Local TFM	Yes
CTAS Terminal[4]	Ground	Research	Air Traffic Control, Local TFM	Yes
FACET[5]	Ground	Research	Strategic TFM	Yes
GE FMS[6]	Air	Operational	Flight Control, Trajectory Planning	Yes
ERAM[7]	Ground	Operational	Air Traffic Control	No

Table 1. Current TP Capability Documents

II. Background

To date, many research and development (R&D) organizations have developed and validated their own individual trajectory prediction technologies to meet the specific needs of their automation concepts and systems of interest. With few exceptions, these efforts have been conducted with little to no leveraging of previous work from other organizations. Over the past decade, several US and European efforts to understand the differences between disparate trajectory predictors met with little success. These efforts attempted to survey and document the current and future DST requirements for trajectory prediction capabilities to identify commonalities and differences. The primary goal of the surveys was to obtain and compile this information in a way that would facilitate cross comparisons, the results of which would point to opportunities to develop “common” TP capabilities to be shared with and leveraged by the R&D community.

The first effort was conducted in 1999 in the form of a 2-day workshop involving senior technical leads from MITRE Corporation’s Center for Advanced Aviation System Development and NASA. The scope was limited to a small set of ATM automation applications, primarily en route, including the Center TRACON Automation System (CTAS),[8] the User Request and Evaluation Tool (URET),[9] and envisioned enhancements to both. The results of the workshop, captured in an annotated briefing, included a side-by-side comparison of the major TP capabilities of each application highlighting similarities and differences. The comparison also included a summary of the major drivers behind the development of the capabilities. This exercise did provide an initial understanding of both the high-level similarities and differences in TP capabilities for these applications, as well as some insights regarding how to compare TP requirements and capabilities. However, the details were at too high of a level to point to significant opportunities for common capabilities. A major impediment was the lack of documentation of the salient details related to each TP.

The second major effort was a joint activity conducted under the auspices of the US-Europe ATM R&D Action Plan 16 (AP16) for Common Trajectory Prediction Capabilities. AP16 is a team of senior TP experts representing the Federal Aviation Administration (FAA), NASA, Eurocontrol research labs, and major industry R&D organizations developing air traffic control and airborne (e.g., avionics and airframe) automation systems. The team conducted a broader survey requesting the details of their TP requirements and capabilities (in any form) from approximately twenty research and operational organizations. The request introduced the survey and described the specific aspects of TP for which information was needed. Key TP-related technical leads were contacted directly through a formal letter of request, email, and follow-up phone calls. After a year of effort, many indicated they had nothing to offer. Of the few that did respond with relevant details, the content varied widely from one organization to another, the material was inconsistent in what was documented (little comparable overlap), and the scope of information was significantly incomplete.

Several key conclusions were generated based on feedback from these past efforts and analysis of the limited material obtained. First, documentation of this type is a systemic challenge for the community, particularly the research labs where the changes in requirements and capabilities can occur quite frequently. Second, the focus of most developers of automation concepts/systems seems to be limited to higher-level requirements for their automation system. Many TP clients have difficulty in specifically determining what their automation concept/system requires from its supporting TP capabilities. Third, the wide range of approaches taken to implement TPs, for research and operational systems, makes it difficult to understand the similarities (and true differences) of two or more TPs. For significant progress in the development and reuse of TP capabilities across the R&D community, a simple and concise methodology is needed to “standardize” the documentation of clear, consistent, complete, and cross-comparable TP requirements and capabilities. A key element of this methodology is the development of a document structure (i.e., the Requirements Framework) that enables effective capture of these TP requirements and capabilities. The documenting and comparison of the five TP technologies described in Table 1 is an initial attempt to develop such a document structure.

III. Lessons Learned

Though a formal comparison of the five TP capabilities documents was not yet scheduled, an opportunity to perform an initial comparison of four of the documents existed because these documents were all being developed by a single team of contractors (L-3 Communications and GE - Aviation). After the completion of early drafts of these four documents, this basic comparison of the documented capabilities was performed. Though the current structure of the Requirements Framework did support this basic comparison, differences in techniques for describing capability details impeded the ability to compare the capabilities. The main result of this initial comparison was the identification of an overarching requirement needed to perform a successful comparison: the TP capabilities documents needed to be written in a more generalized way and less from a TP implementation perspective (i.e., describing how the specific TP capabilities were implemented).

To successfully compare TP capabilities using the original, non-generalized documents, the person performing the comparison was required to abstract or generalize the capability descriptions for each TP to try to identify commonalities and/or differences. The problem was that no single person could be expected to have enough specific knowledge of all of the TPs to effectively perform this abstraction. Therefore, the only realistic approach to performing the comparisons was to have the TP capability document developers themselves rewrite their capability descriptions using abstracted concepts and language. Since these documenters are experts for their specific TP, they would have the required knowledge to successfully perform such an abstraction. An additional benefit of performing this abstraction in the original documents is that the capability documents would now be more easily understood by a wider audience. To maximize the chances that the abstraction of the various TP capabilities by the different TP experts would result in easily cross-comparable capability descriptions, a common set of abstraction techniques to be used by all of the TP experts was developed. These techniques are described next.

IV. Abstraction Techniques

In the following sections, each of the three main abstraction techniques developed is described. The description starts with a discussion of the issue that the specific technique is designed to resolve and then provides a description of the technique itself. For each technique, the resultant benefits to the development of the Requirements Framework and to efforts beyond the documentation and comparison of TP capability documents are also described.

A. Separating Behavioral from Mathematical Modeling

Figure 1 illustrates a conceptualized model for the typical minimum set of inputs and outputs of a trajectory predictor (TP). In the model, the TP accepts aircraft initial condition data, represented by the state input. The state data is typically generated by sensors within the TP client application's operational environment (e.g., radar track data for ground-based systems or GPS data for airborne systems). The TP also accepts a wide range of data that may impact the future path of the aircraft, represented by the intent input. Intent data can include any type of information available to the TP client application and varies greatly based on the client application's operational environment. Examples include: flight plan data, controller issued altitude and time constraints, and aircraft guidance mode settings. Intent data is often equated with constraint data, but any data that can impact the future path of the aircraft is acceptable intent data, including pilot and/or controller preferences and objectives. The final input data is atmospheric data, which is a model of the airspace's winds, temperature and pressure aloft. The output of the TP is a 4-dimensional (4D) trajectory representing the predicted future states of the aircraft. The 4D trajectory output starts at the input state location and predicts an aircraft's future path that is consistent with the input intent data.

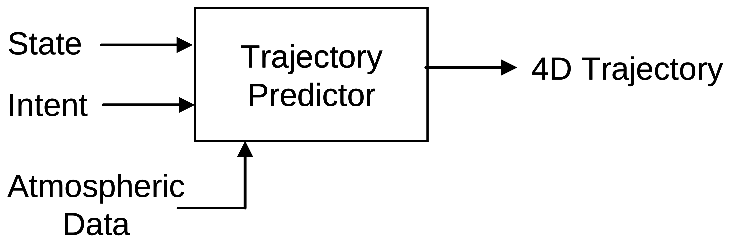


Figure 1. Trajectory Predictor Inputs and Outputs.

It is the responsibility of the TP to convert the state and intent input data into a set of mathematical representations that can be numerically integrated to generate the output 4D trajectory. Each TP implements this process differently, attempting to maximize efficiency and effectiveness given the specific constraints of its application environment (e.g., available input data, implemented mathematical equations of motion, required computational performance). To identify common capabilities between TPs whose implementations vary significantly, a generalized approach for describing the modeling capabilities of a TP was created and used as the basis for abstracting each TP's capabilities. Figure 2 shows this generalized approach, which separates the details of the predicted aircraft maneuvers from the numerical integration approaches used to predict these maneuvers (i.e., generalizing the description of the TP process for turning state and intent data into mathematical representations).

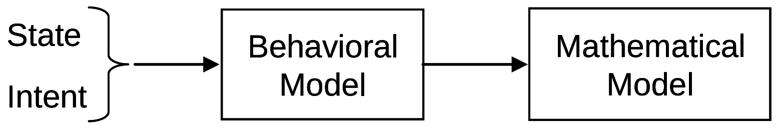


Figure 2. Generalized Approach for Generating the Mathematical Representation.

In Figure 2, the state and intent inputs are first converted into a behavioral model for the aircraft. As described above, the intent represents the known constraints, objectives, etc. of the aircraft. The **behavioral model** is the TP's representation of how the aircraft (pilot, guidance system, etc.) responds to those constraints and objectives. In other words, it is a model of the aircraft's behavior while attempting to meet the constraints and objectives represented in the intent data.[§] The second step is then to convert each element of the behavioral model into a mathematical model. The **mathematical model** defines the mathematical representation of each maneuver defined in the behavioral model; the mathematical model describes how each maneuver will be integrated, including the required equations of motions, integration time step, integration algorithm, etc.

Effective behavioral modeling is done in layers, describing aircraft maneuvers in ever-increasing levels of detail. Figure 3 illustrates an example of vertical behavior modeling for an aircraft whose initial state is not at its cruise altitude. The input data to the TP includes the aircraft's initial altitude (part of the state data) and the cleared cruise altitude (part of the intent data). From this information, the TP models two maneuvers representing the top level (layer) of the aircraft's behavioral model: the aircraft will start in a CLIMB maneuver and transition to a CRUISE maneuver when reaching the cruise altitude at the top of climb (TOC).

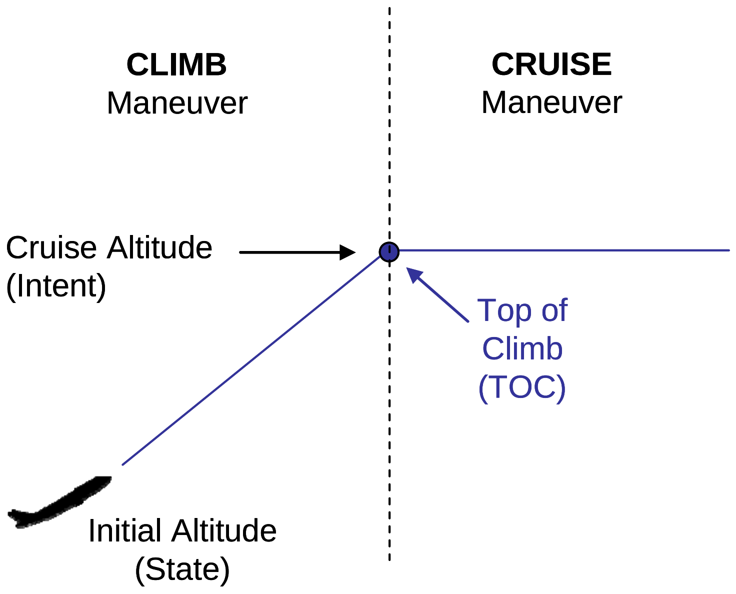


Figure 3. Behavioral Modeling Example: CLIMB Transitioning to CRUISE Maneuver

The TP may require defining the CLIMB and CRUISE maneuver models in more detail (e.g., to achieve its functional and performance requirements). In Figure 4, the CLIMB maneuver has been expanded into a series of sub-maneuvers. The aircraft is now modeled as starting in an IAS CLIMB sub-maneuver, where the aircraft is maintaining maximum climb thrust and an indicated airspeed (IAS). This sub-maneuver transitions into a MACH CLIMB sub-maneuver at the Mach/IAS transition point, where now maximum climb thrust and a Mach number are maintained until reaching the TOC. These two sub-maneuvers could be further broken down into smaller maneuvers if further modeling detail were required. For example, if an acceleration segment from the initial IAS to the target climb IAS was required for the IAS CLIMB sub-maneuver or if a "flare" maneuver near the TOC was required for the IAS Mach maneuver.

[§] The behavior is the result of attempting to achieve the constraints & objectives in the intent because the aircraft may not be able to meet all of these constraints and objectives.

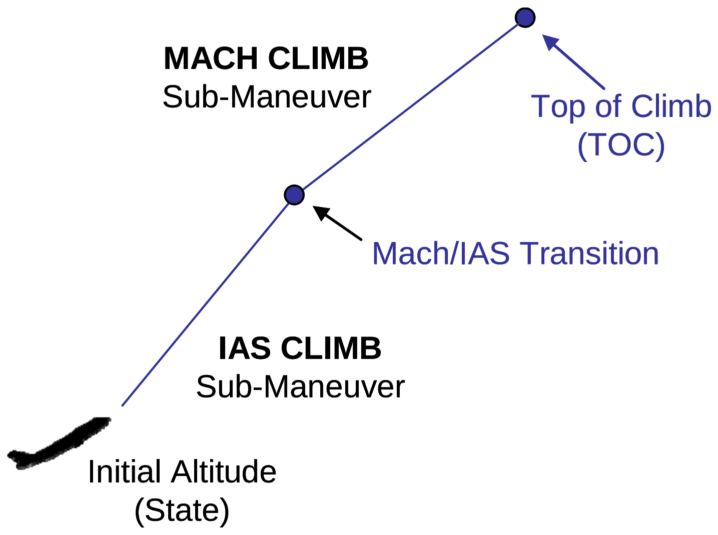


Figure 4. Behavioral Modeling Example: Expanded CLIMB Model

Each maneuver in the top-level behavior model should be broken down into smaller sub-maneuvers until the desired level of maneuver detail is reached. Once this is completed, then a mathematical model can be defined for each of the lowest level sub-maneuvers. Table 2 illustrates a possible final behavioral and mathematical modeling for the CLIMB maneuver. For each mathematical model, a complete set of information must be provided to enable the required numerical integration. Table 2 provides a subset of the required information for a mathematical representation that uses point-mass equations of motion with a kinetic aircraft performance model.[¶]

Behavioral Model		**Mathematical Model**					
Maneuver	*Sub-Maneuver*	*Equations of Motion*	*Speed*	*Aircraft Performance Model*	*Integration Direction*	*Integration Algorithm*	*Timestep*
CLIMB	IAS CLIMB	Point-Mass: Known Speed & Thrust	Constant IAS	Kinetic; Thrust from Max Climb Table	Forward	2nd Order Runga-Kutta	60 seconds
	MACH CLIMB	Point-Mass: Known Speed & Thrust	Constant Mach	Kinetic; Thrust from Max Climb Table	Forward	2nd Order Runga-Kutta	60 seconds

Table 2. Behavioral and Mathematical Models for Example CLIMB Maneuver

1. *Impact on the Requirements Framework*

Several major benefits are achieved by separating behavioral and mathematical modeling in the documentation of capabilities within the Requirements Framework. One benefit is that describing the behavioral modeling of a TP is a direct form of abstraction that supports more effective comparisons. Aircraft behavior is, by definition, independent of any specific TP. The aircraft will perform some series of maneuvers based on its objectives and constraints, independent of whether there are one or more TPs trying to predict its behavior. A TP's behavioral modeling description is therefore directly comparable to any other TP's behavioral modeling, since they are all trying to model the TP independent aircraft behavior. Each TP may model this behavior to a different level of detail, or some TPs may not be able to predict all maneuvers due to a lack of input intent information, but these are exactly the types of differences the comparison is attempting to identify. Differences in mathematical representations are

[¶] An aircraft performance model (APM) is a subset of the mathematical model, providing a mathematical representation of just the aircraft's dynamics. Kinetic APMs typically provide either altitude rate (climb or descent rate) and acceleration data or thrust and drag values from which altitude rates and accelerations can be calculated.

also important, but behavioral modeling capabilities are much more likely to represent potential functional requirements for future TPs. For example, the type and detail of aircraft maneuvers required by the TP are much more likely to be defined as a functional requirement than whether the TP needs to use point-mass equations of motion with a kinetic aircraft performance model (which is more likely to be a design approach to meet a performance requirement).

Another benefit of documenting behavioral modeling separate from mathematical modeling, more specifically through the use of layered modeling descriptions, is the ability to evaluate very complex, detailed behavior modeling logic at different levels of detail. For example, the behavior model can be initially evaluated at only the highest level, as in Figure 3, describing just major maneuvers (climb, cruise, descent, etc.) without the complication of evaluating detailed modeling for each of these maneuvers. This enables TP comparisons to identify which major maneuvers the TPs have in common and which maneuvers are unique to one or the other TP. Then, for each common major maneuver, differences in modeling details can be evaluated. This method for comparing TP behavioral modeling, enabled by the layered description approach, provides additional enhancements to the comparisons:

1. Two TPs with significantly different levels of behavioral modeling can be easily compared
2. Two TPs with complex behavioral and mathematical modeling can clearly identify if differences existed in the maneuvers predicted, the level of detail of the maneuver modeling, or in the mathematical representation of these maneuvers

In the case of two TPs with significantly different levels of behavioral modeling detail, both TPs will typically use the same categories for their major behavior modeling. For all TPs, the major behaviors revolve around the conventional description of aircraft phases of flight: climb, cruise, and descent. If both TPs are modeling a climb maneuver, even if they differ greatly in the level of detail with which they model a climb (e.g., a single climb maneuver versus the two sub-maneuver behavioral model described in Table 2), they both typically identify the major maneuver as a climb. Therefore, the comparison can successfully identify whether (and when) the two TPs predict climb maneuvers. This comparison could be difficult without the layered approach to behavioral modeling since the detailed behavior models and underlying mathematical representations can be very different.

Since the layered approach to behavioral modeling defines maneuvers at a high level and then drills down into lower levels of detail for each maneuver, two TPs with complex behavioral modeling can be compared to identify the level of detail at which the two approaches diverge. The difference may occur at the top-most behavioral modeling level (e.g., one TP does not predict climbs), at some intermediate level of behavioral modeling (e.g., one TP does not model constant IAS and constant Mach sub-maneuvers), or at the mathematical modeling level (e.g., one TP uses a kinetic aircraft performance modeling approach and the other uses a kinematic modeling approach). This ability to compare disparate TP capabilities is enabled by breaking up the complex behavior modeling into smaller, easier to evaluate component behaviors.

2. *Impacts Beyond the Requirements Framework*

The separation of behavioral modeling from mathematical modeling has additional benefits beyond those described above for enhancing the Requirements Framework. First, by describing the behavior modeling independent of the mathematical representation, the assumptions and approximations of the prediction model can be fully evaluated. This is a useful process when designing new TP capabilities (e.g., adding a new maneuver to an existing TP) or when evaluating pre-existing TP capabilities, as in the context of identifying potential modeling improvements to meet required performance requirements during validation. Second, as described above, behavioral model representations of aircraft maneuvers are independent of TP implementation. This makes this representation of predicted aircraft behavior an excellent candidate for transferring data between disparate TPs to enable successful interoperability. Each of these benefits is described in more detail.

<u>Modeling a New Maneuver</u>

Figure 5 and Table 3 illustrate an example of using behavioral and mathematical modeling during the design of a new maneuver to add to an existing TP. In this example, the maneuver being modeled is the aircraft's guidance system response to adding a STEP climb constraint to an FMS route, which causes the aircraft to climb to the new cruise altitude (target altitude) defined in the STEP climb constraint.

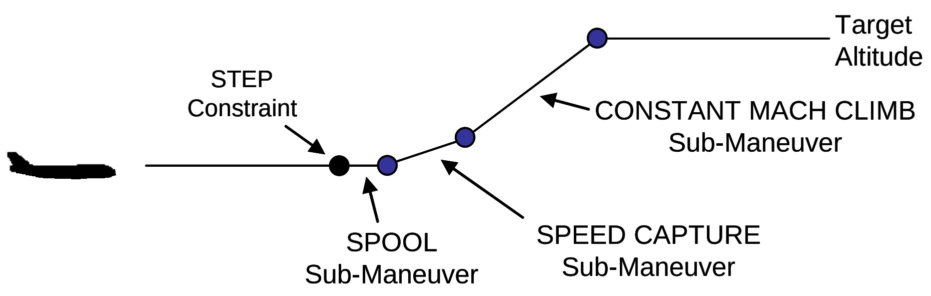


Figure 5. Example of Adding a New Maneuver using Behavioral & Mathematical Modeling.

Behavior Description	**Behavior Model**			**Mathematical Model**
	Maneuver	*Sub-Maneuvers*		
At the step constraint, the autothrottle drives the throttle to maintain maximum climb thrust. The elevator is deflected to capture the target Mach number. Minimum climb rate is 500 fpm.	STEP CLIMB	CAPTURE MACH	SPOOLUP	Constant Altitude and Speed
			SPEED CAPTURE	Constant Vertical Rate (500 fpm) and Max Climb Thrust
After capturing the target Mach number, the Mach number is maintained while climbing at maximum climb thrust.		CONSTANT MACH CLIMB		Constant Mach and Max Climb Thrust

Table 3. Example of Adding a New Maneuver using Behavioral & Mathematical Modeling

The left column of Table 3 describes the (simplified) results of a theoretical discussion with a guidance expert on how the guidance system works when given a STEP climb constraint. Given the behavior description, a high-level STEP CLIMB maneuver can be divided into two sub-maneuvers: CAPTURE MACH and CONSTANT MACH CLIMB. Furthermore, evaluation of the actual aircraft behavior identifies that the finite time for initial throttle and elevator deflection has a negative impact on the predicted maneuver's altitude accuracy if not accounted for in the behavior model, so the CAPTURE MACH sub-maneuver should be divided into two lower-level sub-maneuvers: SPOOLUP and SPEED CAPTURE. Therefore, the definitions for the behavior model's three lowest-level sub-maneuvers are as follows:

- SPOOLUP: Elevator is deflecting based on an undefined control law to capture the target Mach number. The autothrottle is moving to capture maximum climb thrust.
- SPEED CAPTURE: Elevator is deflecting based on an undefined control law to capture the target Mach number. The autothrottle is maintaining maximum climb thrust.
- CONSTANT MACH CLIMB: Speed is being maintained at the target Mach number. Throttle is being maintained at maximum climb thrust.

If it is assumed that the TP uses point-mass equations of motion for all of its mathematical models, then attempting to define the aircraft behavior for the SPOOLUP sub-maneuver at a lower level of detail is not desirable since the ability to model elevator deflection does not exist in the mathematical model. A simplified mathematical model (a simple lag) could be used for this sub-maneuver if the main purpose for this model is to initiate the SPEED CAPTURE sub-maneuver at approximately the correct along-path location (to eliminate the altitude error bias over the entire STEP CLIMB maneuver that exists if the SPEED CAPTURE maneuver is initiated immediately at the STEP constraint location). For the SPEED CAPTURE sub-maneuver, maintaining the throttle at maximum climb thrust can be modeled directly by a kinetic aircraft performance model, but again, the elevator deflection can not be modeled directly. In this case, the mathematical model could use a constant vertical rate at the minimum 500 fpm if this is a close enough approximation of the aircraft's actual vertical rate. The CONSTANT MACH CLIMB sub-maneuver, which should cover the majority of the STEP CLIMB maneuver, can be mathematically modeled directly using point-mass equations of motion and a kinetic aircraft performance model. If during validation the STEP CLIMB maneuver does not meet all of the TP's performance requirements (e.g., altitude prediction accuracy), both the SPOOLUP and SPEED CAPTURE behavioral and mathematical models should first be re-evaluated for acceptable prediction accuracy since they contain the most significant approximations.

TP Interoperability

A critical element of TP interoperability is to share data between systems, both on the ground and in the aircraft, in a way that can be correctly interpreted by the disparate TPs used by these systems. In Europe, the REACT consortium was created to elicit requirements for a standard language to exchange trajectory-related information between ATM systems[10], but efforts similar to this in both Europe and the United States are still attempting to define the data content that can be unambiguously shared between a wide range of ATM systems.

Behavioral modeling, as described above, presents one possible answer to defining this shared data. Because the behavioral model is TP independent, it can safely be shared among disparate TPs without losing its ability to be correctly interpreted. When received by a TP, the behavioral model can be converted into an associated mathematical model that is specific to the receiving TP. Each receiving TP will interpret the behavioral model at a level that is consistent with the mathematical modeling capabilities of that TP. For example, using the STEP CLIMB maneuver described in Figure 5 and Table 3, the behavior model can be described to as low a level of detail as is required by the conceptual objective for sharing the data.[#] Using the theoretical TP from Table 3, if the behavior is modeled to a level that includes elevator deflection and throttle position, this TP will only interpret the behavior model to a level consistent with its point-mass (mathematical) modeling assumptions, ignoring the additional behavioral modeling details. If a second TP of significantly less fidelity (e.g., a TP used for long-range traffic flow management applications) also receives the behavioral modeling data, it may only interpret the behavioral model at the highest STEP CLIMB level. In both cases, the same maneuver has been consistently interpreted by the disparate TPs, which is the required objective. If a third TP were to require data beyond that contained within the shared behavioral model, then this TP would need to model the unknown behavior details based on internal processing, similar to how current day systems perform "intent modeling" when missing critical intent data.[11]

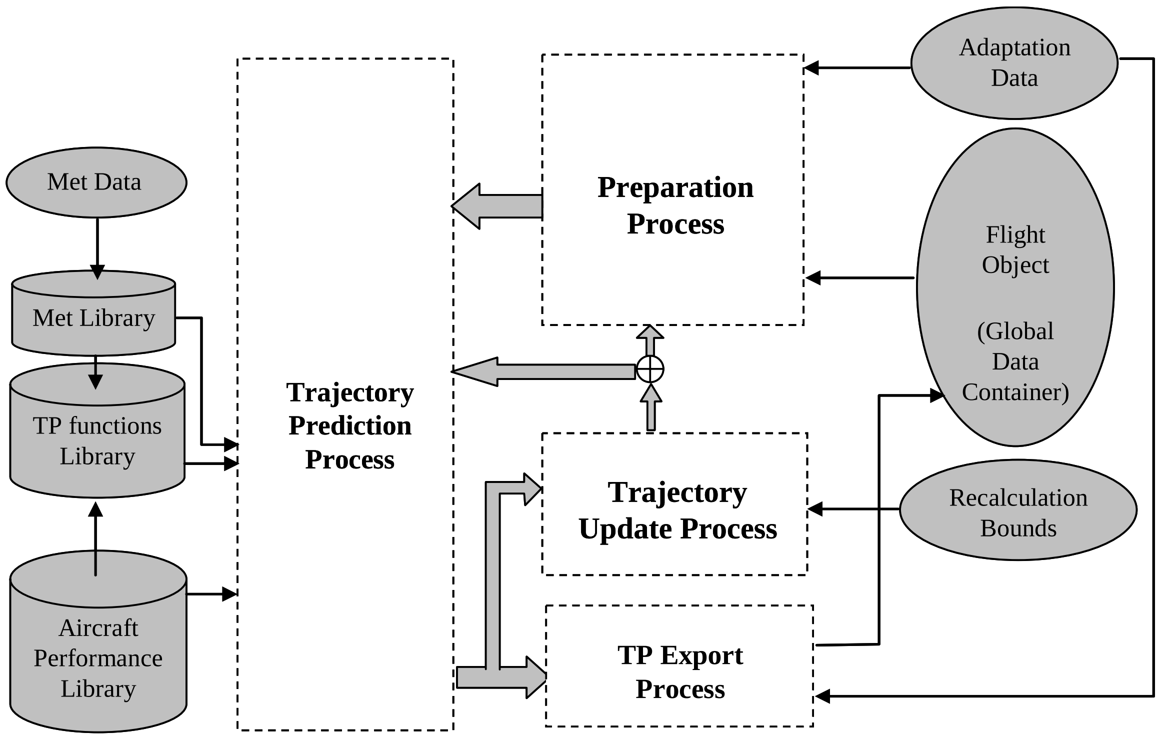
B. Client Application -TP Boundary Definition

Identifying the set of TP capabilities within a given client application would appear to be a straight-forward exercise, but the definition of the boundary between client application and TP capabilities is often blurred through close integration of these capabilities in the application's implementation. For example, a ground-based DST may add an outer loop within the implementation of its "trajectory predictor" that iterates on a trajectory degree-of-freedom (e.g., cruise speed) to meet a time constraint at a lateral waypoint. This implementation may take advantage of the meet-time algorithm's close integration with other TP capabilities to efficiently and effectively find a value for the degree-of-freedom that meets the time constraint, but does this make the meet-time capability a part of the client application's set of TP capabilities? This may at first appear to be an academic question, but an answer to this question is necessary to define the scope of TP capabilities captured within the Requirements Framework and hence, define the ultimate scope of the final comparisons.

The first aspect of the Requirements Framework that defines a conceptual boundary between the TP and its client application is the use the AP16 Common Trajectory Predictor Structure (reproduced in Figure 6) as the Framework's outline.[12] This structure has four process components:

1. Preparation Process – responsible for processing the state, intent and other inputs from the client application into a mathematical representation.
2. Trajectory Prediction Process – responsible for numerical integration of the mathematical representation to create an output 4D trajectory
3. Trajectory Update Process – responsible for internal TP iteration to meet all input constraints
4. TP Export Process – responsible for formatting and processing of the final 4D trajectory before sending to the client application

[#] The required data content will be ATM concept specific, but the approach described should be valid for any level of data content required.



Adaptation Data
Met Data
Preparation Process
Flight Object (Global Data Container)
Met Library
Trajectory Prediction Process
TP functions Library
Trajectory Update Process
Recalculation Bounds
Aircraft Performance Library
TP Export Process

Figure 6. AP16 Common Trajectory Predictor Structure

This structure was developed by the AP16 core team to conceptually cover all potential descriptions of trajectory prediction capabilities, with the full knowledge that the conceptual boundary between the TP and the client application is an on-going subject for debate. The Trajectory Prediction Process, the inner most process in the Common structure, is the one process that is generally accepted as being required within the TP. The other three processes all include some processing that could conceptually be considered either within the TP or within the client application. For example, the meet-time algorithm described above, if considered a TP capability would exist within the Trajectory Update Process. Therefore, the Common TP structure was necessary, but not sufficient for clarifying the client application - TP boundary when documenting the TP capabilities in the Requirements Framework. An additional level of conceptual modeling was required.

A general rule of thumb (referred to as the TP Boundary Rule) was applied to further distinguish, for the sake of documenting the TP capabilities within the Requirements Framework, the conceptual boundary between the client application and TP capabilities: if the capability directly supports a key function of the client application, then the capability is considered a client application capability and outside the scope of the TP. Therefore, for the ground-based meet-time algorithm discussed above, if the algorithm directly provides clearance advisories to the client application user (e.g., a sector controller), then this would be considered outside the scope of the TP since this capability directly supports a function of the client application. The TP Boundary Rule is useful for deciding whether a capability in one of the three “outside” processes of the Common TP Structure (Preparation Process, Trajectory Update Process, and TP Export Process) should be included in or excluded from the TP capabilities description.

The handling of constraint relaxation, a common capability that falls within the grey areas of being considered a TP or client application capability, provides an example of this decision making process. Constraint relaxation is the process of identifying and changing or removing (i.e., relaxing) the requirement to achieve a particular constraint if all of the constraints on the aircraft can not be simultaneously achieved. A classic example of constraint relaxation is when a required descent to an altitude constraint can not be achieved within the current performance limits of the aircraft. Figure 7 illustrates this situation. The identification that the altitude constraint can not be achieved is a capability of the TP since it requires a prediction of the required top-of-descent (TOD) location. The question is whether the relaxation of the altitude constraint should also be considered a TP capability or whether it should be considered a client application capability.

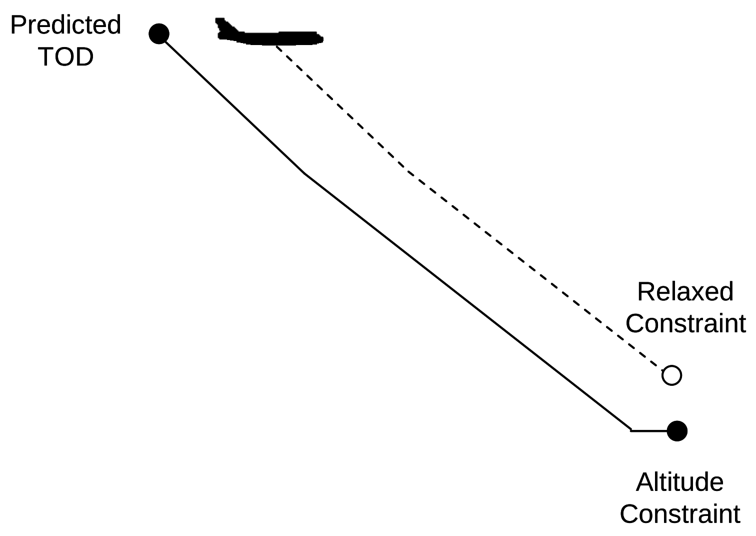


Figure 7. Constraint Relaxation Example

Using the TP Boundary Rule, this capability should be considered a TP capability only if it does not directly support a function of the client application. The following are two possible client application scenarios, one where the relaxation should be considered a client application capability and the other where it should be considered a TP capability:

1. Client Application Capability Scenario: After identification that all constraints cannot be met, an algorithm is performed to identify that the altitude constraint should be relaxed to meet a client application objective (e.g., orderly traffic flow).
2. TP Capability Scenario: After identification that all constraints can not be met, an algorithm runs to modify the pilot procedure to achieve the altitude constraint as closely as possible. The client application displays the amount by which the altitude constraint was not met to the user.

In the client application capability scenario, the identification of which constraint should be relaxed is tied directly to specific client application functionality. Because the selection of the altitude constraint for relaxation is the direct result of obvious client application functionality (specifically, the functionality that creates an orderly traffic flow), this capability is considered part of the client application and not the TP. In the TP capability scenario, no client application functionality requires control over which constraint is relaxed, because the client application only displays which constraint(s) will not be met and by how much based on an assumed pilot procedure. In this scenario, the client application is not using the relaxation of constraints to directly support its functionality, so the capability can be considered part of the TP.

1. *Impact on the Requirements Framework*

Improving the definition of the conceptual boundary between the TP and the client application has significant impacts on the capabilities captured within the Requirements Framework. First, the definition of this boundary directly defines the state, intent and other input data to the TP. This boundary also directly impacts the definition of the capabilities included in the Preparation Process, since this is the process that handles input data. Identification of which capabilities should be included in both the Trajectory Update Process and TP Export Process are also impacted. Because the TP Boundary Rule is simple enough to be consistently applied by the various TP capability documenters, a level of consistency between the TP capability documents was achieved beyond that from just using the Common TP structure as the outline of the Requirements Framework.

2. *Impacts Beyond the Requirements Framework*

The TP Boundary Rule described above does not eliminate all ambiguity in defining what processing should be included in the Common TP Structure. Any capability that directly supports a key function of the client application can be safely considered outside the scope of the TP, but it is not correct that all other capabilities are within the scope of the TP. This can be best illustrated with an example.

Assume that a ground-based DST is predicting the behavior of an aircraft that has just been issued a required-time-of-arrival (RTA) constraint. From the ground-based DST perspective, the aircraft guidance system's algorithm that iterates on FMS cost index to meet the RTA constraint does not have a direct impact on the DST's functionality. So the temptation might be to consider the capability to predict this algorithm's impact on aircraft behavior as a TP capability (i.e., this algorithm is considered part of the Trajectory Update Process in Figure 6). The problem with this conclusion is that including this type of capability inside the TP can, for more complex scenarios, require adding capabilities to the TP that are out of scope for TP processing. For example, take the same scenario, but now assume that the aircraft being predicted is a fully autonomous aircraft that has airborne self-separation assurance (ASAS) equipment on board. This new aircraft would include the ability to create a conflict-free path that meets the RTA constraint, possibly using both lateral and vertical degrees of freedom. The capability to predict this aircraft's behavior would require emulating a complex meet-time algorithm integrated with conflict detection and resolution capabilities. Including this level of capabilities, with the required addition of other aircraft trajectories as TP input data, is outside the scope of trajectory prediction capabilities.

This issue illustrates that the conceptual boundary between the TP and the client application may benefit from an additional layer of processing. As illustrated in Figure 8, a new process (the Trajectory Management Process) is inserted between the client application and the TP process (which includes all of the processes in the Common TP Structure, Figure 6). This new process is responsible for capabilities outside of the client application that predict aircraft behavior, but that are also considered outside of the scope of TP capabilities. This conceptual structure solves the issues identified in the scenario of the ground-based DST attempting to predict the trajectory of an autonomous aircraft with ASAS equipment. In the new structure, all of the aircraft's algorithms associated with creating a conflict free trajectory that meets the RTA constraint, including conflict detection and resolution, can be added to the Trajectory Management Process. This removes the need to include these capabilities within the Common TP structure, while still enabling the client application to consider an Aircraft Prediction Process (i.e., the combination of the Trajectory Management Process and the TP) as separate from the client application.

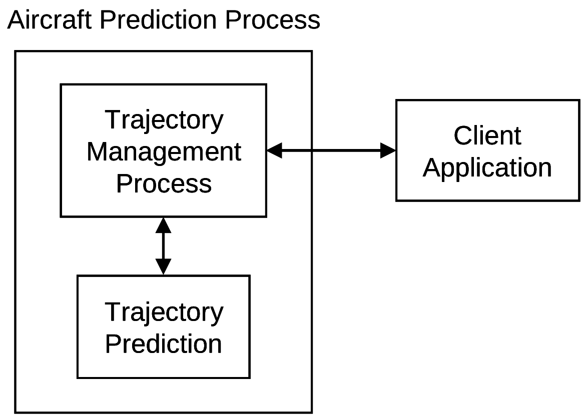


Figure 8. Addition of Trajectory Management Process outside Common TP Structure

C. Use of Abstracted Instead of Domain-Specific Language

An additional impediment to the effective comparison of TP capabilities is when capability descriptions use language that is too closely tied to the domain of the client application. This domain-specific language can cause difficulty in identifying common capabilities, since TPs from very different client application domains (e.g., an airborne TP versus a ground-based TP) can use different names for similar predicted behavior. The use of language specific to the implementation of a TP can cause a similar issue, since TPs can implement the same capability in different ways. The solution to these problems is to use abstracted terminology that generalizes the capability being described.

Figure 9 and Table 4 illustrate this situation by describing a common aircraft maneuver in airborne-specific, ground-specific and abstracted language. The abstracted description describes the series of aircraft maneuvers in domain independent language.[**] The aircraft performs an immediate turn to a pre-defined heading and then follows this heading until capturing (lateral distance within 2 nmi.) a defined route. Once the route is captured, the aircraft

[**] All three descriptions are related to behavioral modeling, but the use of domain independent language is a separate issue from the behavioral modeling discussion above.

turns until capturing a ground track that will provide a constant track path directly to the first turn waypoint in the route ahead of the capture location (note this is a different turn definition than the previous turn). The aircraft maintains this ground track until performing the turn at the first turn waypoint. After this turn, the aircraft follows the remainder of the route (as pictured).

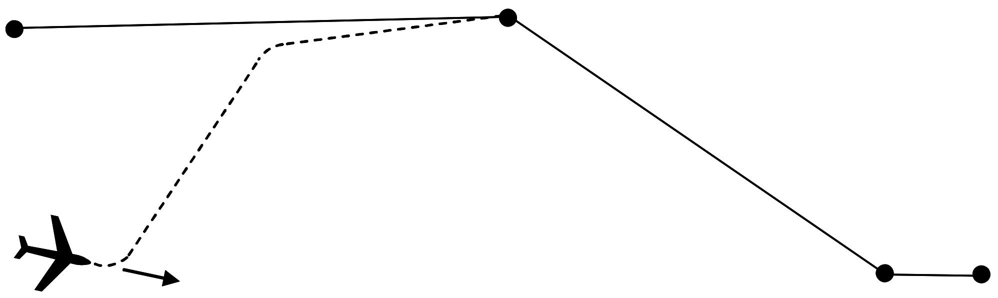


Figure 9. Example of a Lateral Aircraft Maneuver for Illustrating use of Abstracted Language

Abstracted Description	Airborne-Specific Description	Ground-Specific Description
Turn until capturing defined heading	Heading Select mode with target heading (LNAV armed) until within 2 nmi of active route	Turn to cleared heading
Maintain constant heading until getting within 2 nmi. of defined route		Follow cleared heading until within 2 nmi of flight plan route
Turn to track value to capture first waypoint "ahead" of aircraft	LNAV Mode	Follow flight plan route
Maintain track until initiating turn at waypoint		
Turn to capture next segment of route		
Continue following route		

Table 4. Example of Abstracted and Domain-Specific Language for a Lateral Aircraft Maneuver

The airborne- and ground-specific descriptions exhibit both a lack of proper detail and design abstraction in describing the expected maneuvers. The airborne description defines the aircraft's lateral maneuver in terms of airborne guidance mode capabilities. Heading Select is an aircraft guidance mode that will perform a turn until capturing a target heading and then will maintain this heading until the guidance mode is changed. In this case, there is an allusion to the use of "LNAV armed", which is a state of the guidance system that will cause the aircraft to capture and follow the route (in an FMS, this route is called the active route) when the aircraft gets within 2 nmi. of this route. The Lateral Navigation (LNAV) mode follows the route when engaged. Without knowing how airborne guidance systems work, it would be difficult to identify that the abstracted description is equivalent to the airborne description. Similarly, the ground description uses ground-based language, namely the reference to a heading clearance and the following of a flight plan route. The abstracted description not only provides clear language for describing the development of the aircraft's lateral path, it also forces the documenter to identify and describe modeling details (such as the development of the transition from the constant heading segment onto the route via capturing the first waypoint ahead of the aircraft when the route is captured).

1. *Impact on the Requirements Framework*

The major impact on the Requirements Framework of using abstracted language is an increased ability to correctly identify similarities and differences in TP capabilities. An additional benefit is the documenter's enhanced ability to identify an appropriate level of detail to describe the TP capability. As future TP capabilities are documented and compared, an opportunity exists to identify common abstracted terms that can be reused by future capability documents to enhance the ease with which these new capability documents can be compared against existing documents.

2. *Impacts Beyond the Requirements Framework*

The definition of common terms was a major objective of early AP16 activities.[13] The identification of common terms is a critical component of enabling TP experts from different client application domains to discuss and share information about trajectory prediction techniques and technology. The development of abstracted language to enable enhanced comparisons of TP capabilities will provide many abstracted terms that are perfect candidates to be added as common terms to the AP16 list of Common TP terminology.

V. Additional Benefits/Issues Uncovered by Abstraction

A. Library of Common Trajectory Prediction Functions

One of the key insights described above is the decomposition of intent into the basic elements of aircraft behavior and the mathematical representation of those behaviors. Such decomposition allows a broad range of mathematical models to be considered independently from the actual behavior. This approach would enable TP developers to identify and collect a broad variety of mathematical models (or develop new ones) with standardized interfaces. A common set of mathematical models would enable developers to compare and contrast the advantages and disadvantages of specific models for representing the aircraft behaviors that are key to the success of their application. Similarly, developers could identify and collect (or define new) aircraft behaviors that are key to meeting the prediction accuracy requirements for specific applications. The creation of these sets lends itself to the development of common libraries of aircraft behaviors and mathematical-modeling functions. TP developers can then leverage these libraries to improve their trajectory predictor or to create a new one.

A compelling analogy may be drawn from the Object-Oriented (OO) methodology in software engineering where the concept of "abstraction" is fundamental to the methodology. Consider aircraft behavior as a type of "object" representing a set of intended maneuvers. OO development defines what an object is and does before deciding how it should be implemented. Internally, objects are comprised of specific functions that represent how the object responds (its behaviors) to changes in state. Externally, objects are defined by an external interface. Any object can then utilize and build upon other objects simply through the use of the external interface established for each object. This allows the detailed implementation of each object to remain "hidden" and avoids the complexity internal to an object's implementation. Objects (and their internal functions) can then be grouped and maintained within libraries that can be accessed by a broad range of applications.

Aircraft behaviors (trajectory maneuvers and sub-maneuvers) can be viewed in a similar manner, e.g., as objects. Mathematical models correspond to the functional implementation of an object. Libraries of standard aircraft behaviors and mathematical modeling functions could be created and used by one or more TPs supporting a variety of different decision support automation systems and concepts. Standardization would require clear definitions of the inputs to each model function and corresponding outputs.

By separating the behavioral model from the mathematical model, there is a strong potential for reuse of TP modeling functions developed in the future if they are developed with a common interface. Different mathematical models could then be interchanged within a particular behavior model. If a TP had originally been developed to use one particular mathematical modeling approach (e.g., using a kinematic aircraft performance model), the results of using an alternative approach (e.g., a kinetic aircraft performance model) could be analyzed by swapping those functions with others from the library. When selecting the appropriate mathematical models for a new TP application, a developer would consider such factors as the available input data and the required level of prediction accuracy. Lack of input data required to use the particular model needed to achieve required prediction accuracy would demand that additional information be made available. This would provide significant insight into what should be represented in a standard language for trajectory exchange.

With respect to the specific TPs studied in this paper, it is interesting to note that several were developed using object-oriented languages and many of the behaviors were encapsulated within the source code. While an OO implementation of a TP can facilitate the abstraction of aircraft behaviors and corresponding mathematical models, it does not ensure extensibility to a library of common capabilities for multiple TPs. The logic and objects would not be compatible across (and interchangeable between) the separately developed TPs. As such, the functions and objects within the study TPs are not immediately transferable to a standard library. However, to be extensible, a full abstraction of the aircraft behaviors and mathematical models is needed to ensure that they are compatible across (and interchangeable between) multiple TPs. Future TP development activities could benefit significantly from the creation and leveraging of such common capabilities. Potential benefits include reductions in the efforts necessary for TP validation and overall life-cycle costs.

B. Trade-Off between Compatibility of Disparate TPs and Individual TP Prediction Accuracy

The lessons learned from abstraction also lend themselves to help answer an important question related to the definition of TP requirements. While trajectory prediction accuracy is a significant concern for many of the advanced NextGen and SESAR concepts, the system needs for interoperability of TP-based automation systems are equally important. The interoperability of automation systems (e.g., air and ground) is a function of the synchronization of the supporting TPs, among other things. In other words, two different automation systems may develop/advise conflicting decisions if their supporting TPs are not synchronized well enough to result in "similar" predictions (i.e., common situational awareness). The need for synchronization of predicted trajectories can conflict with the TP accuracy needs of any one system. In terms of TP requirements, this leads to the question of which is more important: the compatibility of functional requirements for two or more TPs, or the accuracy of any one TP?

If the TP for one client application has access to data (e.g., weight, thrust, flap schedules) or mathematical models that improve TP accuracy but cannot be shared with another client application/TP for which synchronization is critical, there may be significant differences in the trajectory predictions supporting each client application. This could pose an issue for an operational concept that depends on the sharing of trajectory information. This would be a case where synchronization (resulting in similar trajectory predictions across the two systems) may take precedence over a higher level of modeling accuracy in either system. Here accuracy refers to the degree of conformance between the predicted position and/or velocity of a vehicle (e.g., aircraft) at a given time and its true position and/or velocity.[13] The earlier section discussing TP interoperability explains how the behavior models can be used to define the data that must be shared to achieve synchronization. If one TP does not have the capability to reconcile a particular mathematical model but is generating the trajectory prediction that is critical to the concept, it may require that its mathematical model be used for synchronization. If the higher fidelity mathematical model produces the critical trajectory, it may put a new requirement on the other TP to adopt its capabilities. The question "which client application's trajectory is more correct" is entirely dependant on the operational concept defining the roles of each client application and the context for how each client application's trajectory prediction will be used.

This also illustrates how key functional requirements for a TP for one client application (that must be interoperable with another) can be identified through the cross comparison and analysis of the types of trajectories that must be predicted by both client applications. The level of sameness necessary for the trajectories will dictate the level of sameness in the abstraction of the behavior and mathematical models. This may drive the specification of requirements down to the mathematical model level where critical similarities and differences can be identified. The abstraction process can be used as an aid in this process. A comparison of the detailed modeling for a behavior, such as that done in Table 3, for the individual TPs could identify whether the differences in mathematical models may have a critical effect on compatibility. If this is examined during the early stages of development, it would save time and cost by uncovering potentially hidden requirements.

VI. Concluding Remarks

The abstraction techniques provided several significant benefits. Most importantly for the TP capabilities documentation effort, they improved the ability to perform effective comparisons between documented TP capabilities. This benefit is crucial for meeting the overall objective of defining a Requirements Framework that is clear, consistent, complete and cross-comparable. In addition, the abstraction techniques have led to the development of enhanced TP conceptual modeling and approaches for identifying TP common terms. Finally, additional benefits to the development of common trajectory prediction functions and defining the trade-offs between the compatibility and accuracy of disparate TPs were discovered. These wide-ranging results will support U.S./Europe activities in defining Common TP approaches, including the development of common validation methodologies and defining data sharing requirements for disparate TP interoperability.

The next step in developing the Requirements Framework will be to perform a formal comparison of the five TP capabilities documents from Table 1. The objective of the first comparisons will focus on identifying commonalities and differences between the capabilities described in these documents. Differences will be evaluated to determine whether they represent real differences in TP capabilities or whether the difference is caused by a deficiency in the Requirements Framework. For example, it is possible that a lack of specificity in the Requirements Framework may cause a failure to identify that a particular capability needs to be captured. Another example is when a deficiency in the Requirements Framework causes two documenters to add the same capability to different sections of the Requirements Framework, resulting in an inaccurate comparison. These deficiencies in the Requirements Framework will be identified and resolved until the existing TP capabilities documents can be clearly and consistently cross-compared.

After completing the comparisons of the existing TP capabilities documents, additional TP technologies from a wider range of operational domains will be targeted for documentation and comparison with the original set of documents. The application to a wider range of TP technologies should help to continue the development of the Requirements Framework until it is sufficiently broad in its scope to cover the desired range of future TP requirements and capabilities. Ultimately, the Requirements Framework will be applied to define requirements for a new, advanced TP being developed for one of the future ATM concepts being proposed in the United States and Europe. Through the successful analysis of existing TP capabilities, the result should be a clear, consistent, complete and cross-comparable documentation of these TPs requirements.

VII. Acknowledgements

The authors would like to express gratitude to the developers of the TP capabilities documents, David Karr (L-3), Vincent Kuo (L-3) and Joel Klooster (GE - Aviation), for all of their input during the development of the abstraction techniques. The authors would also like to thank Lockheed Martin, in particular Rich Smolen, Sergio Torres, and Ed McKay, for their continued efforts in supporting the development of the Requirements Framework. Finally, the authors would like to recognize their colleagues in the AP16 group for all of their efforts in the advancement of Trajectory Prediction technology, in support of which this effort was conducted.

VIII. References

[1]Joint Planning and Development Office, “Concept of Operations for the Next Generation Air Transportation System,” Version 2.0, 2007.

[2]SESAR Consortium, “The ATM Target of Operations,” Technical Report No. DLT0612-001-02-00, Toulouse, France, 2007.

[3]L-3 Communications, “Center-TRACON Automation System (CTAS) En Route Trajectory Predictor Requirements and Capabilities”, Version 1.10.

[4]L-3 Communications, “Center-TRACON Automation System (CTAS) Terminal Area Trajectory Predictor Requirements and Capabilities”, Version 1.0.

[5]L-3 Communications, “Future Air Traffic Management Concepts Evaluation Tool (FACET) Trajectory Predictor Requirements and Capabilities”, Version 1.0.

[6]GE-Aviation, “GE-Aviation Flight Management System Trajectory Predictor Requirements and Capabilities”, Version 1.20.

[7]Lockheed Martin, “En Route Automation Modernization (ERAM) Trajectory Predictor Algorithm Description and Requirements Framework”, Version 1.0.

[8]Erzberger, H., Davis, T. J., and Green, S. M., "Design of Center-TRACON Automation System," *AGARD Meeting on Machine Intelligence in Air Traffic Management*, Berlin, Germany, 11-14 May 1993.

[9]Brudnicki, D. J., and McFarland, A. L., “User Request Evaluation Tool (URET) Conflict Probe Performance and Benefits Assessment”, *Air Traffic Management R&D Seminar*, Saclay, France, 1997.

[10]Lopez-Leones, J., Vilaplana, M., Gallo, E., Navarro, F., and Querejeta, C., “The Aircraft Intent Description Language: A Key Enabler for Air-Ground Synchronization in Trajectory-Based Operations,” *Digital Avionics Systems Conference*, Dallas, TX, 2007.

[11]Vivona, R., “Applications of User Intent in Cockpit- and Ground-Based Decision Support Tools,” *AIAA Guidance, Navigation and Control Conference,* Monterey, CA, 2002.

[12]FAA/Eurocontrol Cooperative R&D Action Plan 16, “Common Trajectory Prediction Capability, Common Trajectory Predictor Structure,” Version 2.2, 2004.

[13]FAA/Eurocontrol Cooperative R&D Action Plan 16, “Common Trajectory Prediction Capability, Common Trajectory Prediction-Related Terminology,” Version 2.0, 2004.