SAND2015-3487

SANDIA REPORT

SAND # Pending
Unlimited Release
Printed May 2015

The Capability Portfolio Analysis Tool (CPAT): A Mixed Integer Linear Programming Formulation for Fleet Modernization Analysis

Stephen M. Henry, Frank M. Muldoon, Matthew J. Hoffman,
Gio K. Kao, Craig R. Lawton, Darryl J. Melander, Liliana Shelton

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@adonis.osti.gov
Online ordering:	http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.fedworld.gov
Online ordering:	http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND # Pending
Unlimited Release
Printed May 2015

The Capability Portfolio Analysis Tool (CPAT): A Mixed Integer Linear Programming Formulation for Fleet Modernization Analysis

Stephen M. Henry[1], Frank M. Muldoon[1], Matthew J. Hoffman[1], Gio K. Kao[2], Craig R. Lawton[1], Darryl J. Melander[3], Roy E. Rice[4], Liliana Shelton[5]

[1] Systems Readiness and Sustainment Technologies

[2] Networked System Survivability & Assurance

[3] Software Systems R&D

[5] System Surety Engineering I

Sandia National Laboratories

P.O. Box 5800, Albuquerque, New Mexico 87185

[4]Teledyne Brown Engineering

300 Sparkman Drive, Huntsville, AL 35805-1912

Abstract

In order to effectively plan the management and modernization of its large and diverse fleet of vehicles, the Program Executive Office Ground Combat Systems (PEO GCS) commissioned the development of a large-scale portfolio planning optimization tool. This software, the Capability Portfolio Analysis Tool (CPAT), creates a detailed schedule that optimally prioritizes the modernization or replacement of vehicles within the fleet - respecting numerous business rules associated with fleet structure, budgets, industrial base, research and testing, etc., while maximizing overall fleet performance through time. This paper contains a thorough documentation of the terminology, parameters, variables, and constraints that comprise the fleet management mixed integer linear programming (MILP) mathematical formulation.

This page intentionally left blank.

Contents

Preface 9

Nomenclature 11

1 Business Rules 15

System Transition Flow . . . 15
Mission Priority Tiers . . . 16
Transition Delays . . . 17
General Scheduling Rules . . . 17
Budgets . . . 18
Product Families . . . 19
Low-Rate Initial Production . . . 21
Future Programs . . . 21

2 Formulation Indices, Sets and Tuples 23

Formulation Indices . . . 23
Useful Sets and Tuples . . . 24
Fleet Structure and Flow . . . 24
Optimization Tiers . . . 25
Modernization Scheduling . . . 25
Product Families . . . 26
Future Programs . . . 27

3 Model Input Parameters 29

Optimization Tiers and Phases 29
Fleet Structure and Flow 30
Modernization Scheduling 31
Cost & Budgets 32
Future Programs 34
Auxiliary Parameters 34

4 MILP Decision Variables **37**
Non-Negative Integer Variables 37
Binary Variables 38
Continuous “Binary” Variables 39
Non-negative Continuous Variables 39

5 MILP Variable Expressions **41**
Fleet Structure and Flow 41
Low-Rate Initial Production 44
Storage 44
Cost and Budgets 46
Production 49
Objective Function 50

6 MILP Constraints **53**
Multi-Tier, Multi-Phase Constraints 53
System Flow Conservation Constraints 55
General Scheduling Constraints 57
Budget Constraints 60
Group Density Levels 62
System Production Constraints 63

Product Family Constraints 64

LRIP Constraints 68

Production Smoothing Constraints 69

Future Program Constraints 70

List of Figures

1 CPAT Fleet and Storage Structure 11

Preface

Program executives face the perpetual fleet management challenge of devising investment strategies to assure optimal fleet modernization and to mitigate system obsolescence. These investment plans must be comprehensive, ensuring a balance between capability, schedule and cost. This is particularly true for the Ground Combat Systems (GCS) fleet within the United States Army. Here, capability requirements must be met without violating increasingly strict expenditure limits, which are made in various categories including procurement, recapitalization, operations & support (O&S), and research, development, testing & evaluation (RDT&E). In addition to these demanding budgetary considerations, secondary effects on the industrial base must be carefully integrated, along with numerous other business rules associated with the GCS fleet. This paper presents a mixed-integer linear programming model that helps decision-makers create and evaluate real-world fleet-wide modernization plans. While comprehensive in scope, this paper's concentration on the mathematical formulation itself may fail to elucidate more general modeling approaches, assumptions, and thought processes. Hence, this document should be read in conjunction with the CPAT Domain Model, which presents the CPAT methodology from the perspective of an outside analyst not possessing intimate knowledge of the mathematical formulation.

This page intentionally left blank.

Nomenclature

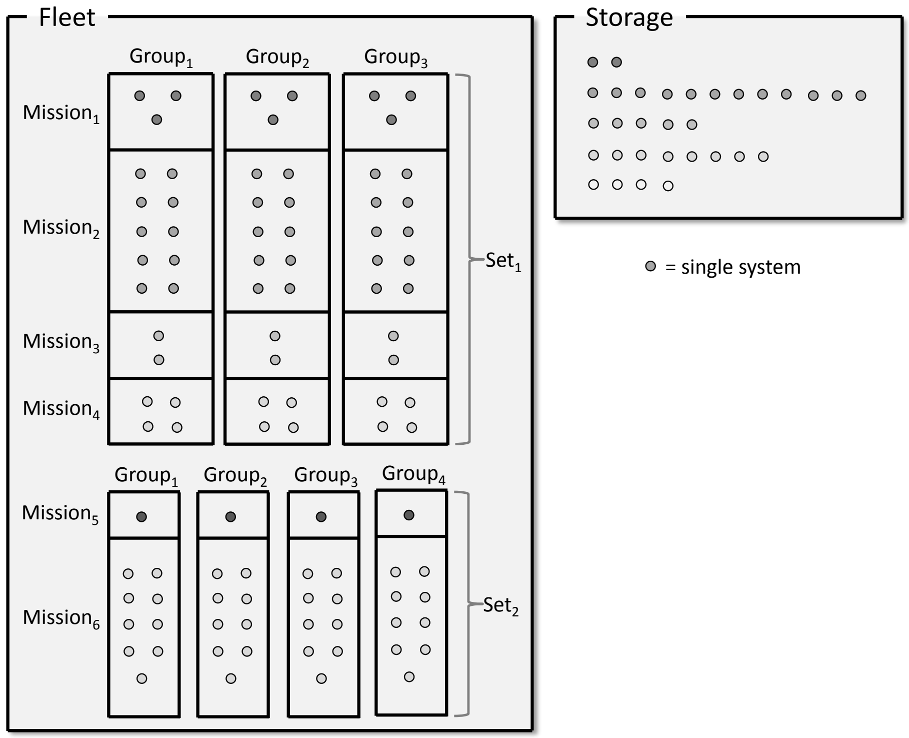

Figure 1. CPAT Fleet and Storage Structure

Fleet: The regimented collection of systems whose performance contributes to the optimization's objective function (i.e. the systems that are actively fulfilling mission roles). The fleet's composition is altered through time as new systems are introduced and old systems are taken out, though the size of the fleet is always constant. Systems in storage are *not* considered part of the fleet.

Set: A high-level partition of the fleet. Each set is itself partitioned into equal groups, and each set's partitioning can be unique.

Group: The equi-partitioning of a set. Each group is itself partitioned into heterogeneous missions. Each group within a set is partitioned identically by mission. In Army parlance these groups are are called "brigades," and we will often use this term interchangeably.

Mission: The partitioning of a brigade into areas of unique operational responsibility. Every brigade within a set is partitioned in an identical manner by mission. The lowest level fleet modernization decisions made by CPAT are for a mission within a group, and often when this document refers to a "group" or a "mission" being modernized, it is meant as a shorthand for "a mission within a group."

System: A resource that fulfills a mission with a certain performance level. Systems are the individual components which are being upgraded, purchased, and swapped within the fleet by the CPAT optimization algorithm.

Storage: A conceptual holding area for systems which are not in the fleet. Systems can flow into and out of storage via a variety of mechanisms.

Transition/Fielded/Modernize: General terms referring to any substitution event that alters the composition of the fleet (i.e. one system type is switched over to another type within a mission). This conversion may happen via a mission upgrade or a storage swap.

Mission Upgrade: A transition that occurs within the context of a mission wherein the original system is consumed in the creation of the new system. While the upgrade is in progress, the mission still "gets credit" for the performance of the original system. Upon completion, the new system is delivered to the mission.

Storage Swap: A transition that occurs within the context of both storage and a mission. Here the original system in the mission is taken out and placed in storage. The new system is taken out of storage and placed in the mission. Note that storage swaps are free and instantaneous, since the new system is already in storage and waiting to be used. However, the process of getting that new system in storage is most likely neither free nor instantaneous (i.e. via a purchase or storage upgrade).

Purchase: The acquisition of a new system that previously did not exist. Newly purchased systems are placed into storage and are immediately available for introduction into the fleet via a storage swap. Purchases are not considered transitions, since they do not directly alter the composition of the fleet.

Storage Upgrade: An upgrade that happens in storage (outside the context of the fleet). Here the original system in storage is consumed in the creation of the new system which is placed in storage and immediately available to be fielded to the fleet via a storage swap. Note that storage upgrades are not categorized as a transition, since they do not directly alter fleet composition.

Delivery: The completion of production (as defined by the admin and/or production delays) for a system. When a mission upgrade completes production, the new system is delivered to the mission. When a purchase or storage upgrade completes production, the new system is delivered to storage. Depending on user specifications, when systems are produced by an LRIP profile, some of the produced systems may be delivered to storage, while others may disappear (destroyed in testing, for example) and never enter storage.

Fielding: The act of placing systems into a mission.

Spoken For: A system is "spoken for" if it is in the process of upgrading from one type to another. More specifically, systems are spoken for when they are in the production period(s) of an upgrade; they are not spoken for during administration periods. We use this notion because systems are still considered in mission or in storage throughout the upgrade process. Hence, we need a way to delineate which systems are already being worked on and which ones are not.

Procurement: A term referring to expenses incurred in the process of modernizing systems. Upgrade, purchase, LRIP, product family start-up and per-period costs all fall under this category.

Product Family: A collection of system types that share production costs, RDT&E costs, and/or resources.

Conventional Time Horizon: Any time period t where $t \leq T$. Highest fidelity decisions are made during the conventional planning horizon.

Extended Time Horizon: Any time period t where $T < t \leq \mathcal{T}$. Lower fidelity decisions regarding future programs are made during the extended planning horizon.

Chapter 1

Business Rules

Before presenting details of the mathematical formulation itself, we first outline the set of business rules that govern the behavior of our fleet modernization model. In outlining these rules, and the formulation at large, many terms will have a specific meaning that facilitates ease of interpretation. See the Nomenclature section at the beginning of this document for a list of common vocabulary conventions.

The business rules below each correspond to a set of parameters, constraints, and expressions within the optimization model. However, some constraints or expressions play an auxiliary role not directly corresponding to a specific business rule; others may address multiple business rules simultaneously.

System Transition Flow

- **Constant System Population:** Throughout the planning horizon, each mission within a group always maintains a constant number of systems. Every change to the fleet consists of either modifying existing systems or removing some number of systems from the fleet and putting an equal number of different systems in their place.

- **Group Purity:** At any given time, the systems serving a particular mission within a particular group must all be of the same system type. Different groups can each be using a different system type for that mission, and different missions within the same group may be using different system types, but a single group cannot mix system types within a mission.

- **Outflow Availability:** For any time period, the number of systems of a given type in a mission that are upgraded or swapped to storage may not exceed the number currently exchangeable. Similarly, the number of systems of a given type in storage that are upgraded or sent to a mission may not exceed the number currently exchangeable. In both cases, the number currently exchangeable is given by the current number present minus the current number in the process of being upgraded.

- **Initial Populations:** Each mission has an initial population of systems that is already in the fleet and is immediately available to begin modernization. There may also be an initial population of systems in storage which is also immediately available to begin upgrading or swapping into missions. In other words, no initial systems are "spoken for" in the first time period.

- **Storage Flow:** Systems enter and exit storage through the following means: 1) purchases put new systems directly into storage, 2) storage upgrades take one system type already in storage and turn it into another type, and 3) storage swaps take one system type out of a mission and into storage while taking another type out of storage and sending it to the mission. Once in storage, a system is immediately available for any type of flow action with one exception: a system cannot be swapped into and out of the same mission in the same time period.

- **No Pre-Usage Upgrades:** Newly purchased systems in storage that have not yet been sent to a mission should not be upgraded while in storage.

- **Optional Pre-Purchasing:** Systems may be purchased or in-storage upgraded before they are actually needed to be fielded to a mission. However, this ability is optional and may be disallowed by user choice.

- **Delivery Implies Fielding:** System types whose procurement cost is non-zero can only be produced if they are also fielded to a mission. (Note that delivery of these systems from production and fielding to a mission can occur at different times.) Only systems that can be procured for free (usually hull systems) can be delivered without also being fielded.

- **No Retire and Re-Fielding:** Systems that are retired from a mission and sent to storage cannot be immediately sent back into that same mission during that same time period.

- **1-Year Duty Minimum:** Systems in a mission must remain for at least 1 time period before they can be swapped out or spoken for by a mission upgrade.

Mission Priority Tiers

- **Priority Tiers:** Fleet missions may be partitioned into priority tiers wherein each tier comprises a separate optimization. The modernization of missions in the highest priority tier is performed first, with subsequent tiers being modernized separately with the remaining budget. Note that all other business rules must hold *in toto* across all tiers. For example, if a product family disallows production gaps, then it may only be started up once even if it fields systems to missions across multiple tiers; it is not allowed to start up separately for each tier.

- **Tier Phases:** Within each tier, there are four separate optimization phases. The first minimizes schedules violation; the second minimizes budget violation while not allowing schedule violations to increase; the third maximizes fleet performance while not allowing either schedule or budget violations to increase; the fourth minimizes cumulative combined fleet costs while preserving fleet performance and not allowing schedule or budget violations to increase. This ensures that 1) if business rules must be broken then budget violations are always preferred to schedule violations, 2) performance is achieved via the most intelligent possible allocation of budget resources and 3) lower tiers, which use the left-over budget from higher tiers, will have the best possible opportunity for modernization.

- **Mission Succession:** One mission can be designated to succeed another so that nothing can be fielded to the succeeding mission until the preceding mission has 1) completely finished fielding and 2) modernized 100% of its original systems. This is typically used for corresponding missions in different tiers (i.e. the IFV mission in the ABCT set must be fully upgraded before the IFV mission in the National Guard set), but can also be used for missions within the same tier.

Transition Delays

- **Delay Partitioning:** When upgrading from one system to another (whether in a mission or in storage) or purchasing a new system, there may be a delay between when the new system is paid for and when it is delivered. This delay is partitioned into an administrative delay (where the system has been paid for but is not yet in production) followed by a production delay (where the system is in production but is not yet delivered). These delays must be accounted for. Default administrative and production delay = 0 periods.

- **Upgrade Admin Delays:** For any upgrade having admin and production delays, the admin period is allowed to begin even if the seed system is not yet on hand. However, the seed system <u>must</u> be on hand in order to begin the first production period. Intuitively, this means that "upgrade paperwork" (i.e. the admin period) can be started in anticipation of the soon to arrive system. Stated another way, while in admin periods a system is not yet "spoken for."

General Scheduling Rules

- **System Modernization Requirements:** System types in the initial fleet may require that a certain percentage be transitioned out before a specified time period. This modernization must be performed. Default requirement = 0%.

- **System Mandates:** Missions may mandate that a minimum number of a particular system type be in that mission during the final time period. This minimum must be met. Default minimum = 0.

- **Per-Period Mission Modernization Limit:** Missions may place an upper bound on the number of groups that can modernized per period. These upper bounds must be respected. Default bound = unlimited.

- **Cumulative Mission Modernization Limit:** Missions may place an upper bound on the cumulative number of groups of initial systems that can ever modernized. These upper bounds must be respected. Default bound = unlimited.

- **Minimum Group Transition Density:** Missions may require that if a transition occurs, then it must occur for at least a certain number of groups. These transition requirements may be specified for up to 3 density levels, which operate as follows:
  Levels = $\{12, -, -\}$ implies that a transition must occur for at least 12 groups.
  Levels = $\{12, 16, -\}$ implies that a transition must occur for at exactly 12 or at least 16 groups.
  Levels = $\{12, 16, 20\}$ implies that a transition must occur for exactly 12 or exactly 16 or at least 20 groups.
  Default Levels = $\{-, -, -\}$.

- **Minimum Group Final Density:** Missions may require that the number of groups of non-initial systems in the mission during the final time period meet certain densities. These densities may be specified by up to 3 levels, which operate analogously to the Minimum Group Transition Densities. Default Levels = $\{-, -, -\}$.

- **System Obviation:** A system type may be obviated by any other system type so that the obviated system can only be delivered prior to any deliveries of the obviating system.

- **Synchronization Sets:** A collection of missions and systems within those missions may be required to modernize or divest simultaneously. For example, if mission M_1 uses system S_1 and M_2 uses S_2, and additionally these missions and systems are part of a sync set, then the number of groups of S_1 entering or exiting M_1 must equal the number of groups of S_2 entering or exiting M_2 for all time periods.

- **Storage Consumption Priority:** Certain systems in storage may take consumption priority over certain other systems. This means that if the higher priority system is exchangeable in storage, then it must be used as an upgrade seed before the lower priority system can be used as an upgrade seed.

- **Upgrades Trump Purchases:** For some systems, modernization must be accomplished via upgrades, if possible. A new purchase is allowed only if no seeds systems are available for the upgrade.

Budgets

- **Per-Period Budgets:** The amount of money spent each period in the 3 categories of Procurement, O&S, and RDT&E must not violate associated per-period budgets for these expense types. Furthermore, a user-specified combination of these 3 per-period budget types must not violate a per-period combined budget. These budgets must be respected by both future and non-future system expenditures throughtout the conventional and extended time horizons. Default budgets = unlimited.

- **Cumulative Budgets:** The total amount of money spent throughout the conventional plus extended time horizon in the 3 categories of Procurement, O&S, and RDT&E must not violate associated cumulative budgets for these expense types. Furthermore, a user-specified combination of these 3 budget types (matching the per-period budget combination) must not violate a combined cumulative budget. These budgets must be respected by both future and non-future system expenditures. Default budgets = unlimited.

- **Early/Late Transition Charging:** No transition may take place in a time period early enough so that associated costs (whether transition, long lead, or product family start-up costs) would be incurred prior to the start of the time horizon. Similarly, no transition may occur in time periods late enough that associate product family start-up costs would be incurred after the end of the time horizon.

- **Long Lead:** Some system types may have long lead on their procurement. This means that a certain percentage of their procurement cost is incurred one year earlier than normal. (Remember that normally procurement costs are incurred during the first admin period.)

Product Families

- **Active Product Families:** Multiple system types can be clustered together into a single product family, with the interpretation that these systems share production facilities and/or RDT&E efforts. A product family is considered "active" (thus incurring per-period costs) during a time period if any member systems are 1) in administrative delay, 2) in production delay, or 3) being delivered and the production delay is 0. Note that both low-rate initial production (LRIP) and full-rate production (FRP) count towards these three conditions, even if the LRIP is being incurred for a separate product family.

- **Family Start-Up Costs:** Each product family may have an associated start-up cost profile that must be incurred when the family first begins work for full-rate production. That is, when the family is 1) in administrative delay, 2) in production delay, or 3) being delivered and the production delay is 0 for the first non-LRIP systems. These costs are allocated to missions using a vehicle density weighting method (see the CPAT Domain Model document for more info.) Default start-up cost = $0.

- **Family Per-Period Costs:** Each product family may have an associated per-period cost that must be incurred every time period that the family is active. Note that a family is active even if its member systems are being produced for LRIP of another family. Like Family Start-Up Costs, these costs are allocated to missions using a vehicle density weighting method. Default per-period cost = $0.

- **Family Per-Period Capacity:** For each product family and time period, there may be an upper limit on the number of member systems delivered during that period.

These limits must be respected, although LRIP does not count towards this capacity. Default capacity = unlimited.

- **Family Cumulative Capacity:** For each product family, there may be an upper limit on the cumulative number of member systems that are ever delivered from the family. These limits must be respected, although LRIP does not count towards this capacity. Default capacity = unlimited.

- **Minimum Sustaining Rate:** Given that systems are delivered from a product family in a particular time period, there may be a lower bound on the number of systems that must be delivered from that family in that time period. These bounds must be met, although LRIP does not count towards this bound. Also, these bounds are not enforced during the last production period, allowing the production line to wind down. Default MSR = 0.

- **Delivery Gaps:** Product families may be restricted so that delivery begins at most 1 time; it cannot start delivering systems, stop, and then subsequently restart. This means that all systems within that family must be delivered during a collection of contiguous time periods.

- **Production Smoothing:** For each product family, there may be a limit on the variation in number of system delivered from that family when in full-rate production. This prevents undesirable effects to the manufacturer. Note that in the final period of full-rate production, this restriction is less stringent so that the production line can begin to wind down output. Default production variation = unlimited.

- **Production Ramp-Up:** For each product family, there may be a ramp-up period prior to full-rate production. During this ramp-up, delivery output is not required to respect production smoothing. Instead, the number of systems delivered must be non-decreasing in time during this ramp-up.

- **RDT&E Costs:** For each product family, there may be an RDT&E cost and systems from the family can be delivered if and only if the RDT&E cost profile of the family is incurred. Default cost = \$0. The analyst may choose to allow the optimization engine to delay certain RDT&E costs in order to avoid budgetary bottlenecks. For each time period that a cost profile is delayed, a separate cost profile must be supplied; a delay (including $d = 0$) is valid only if it has an associated cost profile. Incurring a delay of d time periods also delays the availability of systems in the product family by d time periods. In addition, if $d > 0$, then at least one system within that family must also be delayed by exactly d (other systems may be delayed by more). The analyst may choose to enable legacy RDT&E cost behavior. As before, systems from the product family with an RDT&E cost profile may only be produced if and only if the cost profile is incurred. However, the $d = 0$ cost profile is incurred regardless of when the associated systems are first delivered.

Low-Rate Initial Production

- **LRIP Profiles:** Some systems in some product families may require a modest number of systems be produced in the years leading up to full-rate production for the family. These LRIP profiles define fixed amounts of systems that must be produced up to 5 years before FRP begins. These LRIP profiles have 3 additional analyst-defined properties: 1) not all of the LRIP systems produced have to be delivered to storage (some may be destroyed, for instance), 2) the seed system for the LRIP production may or may not be explicitly defined and, 3) if the seed system is defined, these seeds may or may not be extracted from storage when the LRIP profile is produced.

- **LRIP Timing:** All LRIP profiles incurred by a product family must be lined up so that their final LRIP delivery come exactly one time period prior to the first non-LRIP (i.e. FRP) delivery for the family.

Future Programs

- **Future Program Activation:** Systems that might enter the fleet far in the future can be grouped together into future programs. Future programs are incorporated into the fleet via simple go/no-go decisions. If a future program is activated, then at least one future system associated with the program must be activated. Optionally, each future program may be restricted so that its activation requires that <u>all</u> of its associated systems must be fielded.

- **Future System Fielding:** When a future system is activated, it must be fielded to its mission according to a fixed, user-defined fielding schedule. Optionally, each future system may be mandated to be fielded, in which case the schedule phase would be infeasible were the system not activated.

- **Future Obviates Present:** Once a future system starts fielding to a mission, no other "non-future" systems may be fielded to that mission.

This page intentionally left blank.

Chapter 2

Formulation Indices, Sets and Tuples

Formulation Indices

The following indices are used consistently throughout the formulation for indexing input parameters and decision variables:

- i and j denote system types, of which there are approximately 90 due to the breakout of mission specific variants.
- m denotes missions, which are predefined assignments performed by specific systems with specific performance qualities. Currently, there are approximately 30 missions.
- t denotes time periods (years) in the planning horizon, which currently spans years between FY15 and FY50. Note that time periods are partitioned into the <u>conventional</u> and <u>extended</u> time horizons. The extended time horizon allows for the inclusions of future programs.
- T denotes the ending period of the <u>conventional</u> planning horizon.
- $\mathcal{T}$ denotes the ending period of the <u>extended</u> planning horizon, which spans $T < t \leq \mathcal{T}$. Note, if there does not exist an extended time horizon in the model then $\mathcal{T} = T$.
- p denotes the product families, of which there are currently around 25. Each product family is associated with a start-up cost profile, a per-period cost, a set of RDT&E cost profiles, and a subset of system types. The start-up cost profile and an RDT&E profile must be incurred in order to begin producing systems from the family; the per-period cost must be incurred for every time period that the family is active.
- d denotes the possible number of time periods by which RDT&E costs associated with a product family (and thus the availability of the associated systems) may be delayed. A value of $d = 0$ indicates that the product family starts "on time" while a value of $d = 2$ indicates that the product family starts two years late. Delays of $d \in \{0, \ldots, T-1\}$ are only valid if there is an associated cost profile explicitly input for that delay.
- s denotes the synchronization sets. Each sync set is associated with a collection of missions (given by the set $SyncMissions_s$) and systems (given by $SyncSystems_s$) for

which the number of groups of synchronized systems entering or exiting the synchronized missions must be equal for all time periods.

- $\mathcal{F}$ denotes the future programs, of which there are about 5. Each future program has associated costs (predefined startup and active profiles) along with associated future systems having fixed fielding profiles. Future programs are included in the fleet via simple go/no-go decisions.

- $\mathcal{J}$ denotes the future systems, of which there are about 15. These systems are fielded according to a fixed profile, and can only be fielded if their parent future program has been activated.

Useful Sets and Tuples

Fleet Structure and Flow

- $Roles$ defines the valid pairings (i, m) where system i can serve mission m.

- $InterimRoles$ defines the roles (i, m) where system i can serve mission m where i is an interim system (i.e. neither initial nor final) in mission m. Here $InterimRoles \subseteq Roles$ and is used to filter the $bInterimBrigCanField_{i,m,t}$ variables down to the minimal number needed.

- $MandatedRoles$ defines the roles (i, m) for which there exists a requirement that at least a certain number (given by $FinalMandate_{i,m}$) of systems i must exist in mission m at the end of the conventional time horizon (time T). Here $MandatedRoles \subseteq Roles$ and is used to filter the $iFinalMandateDeficit_{i,m}$ variables down to the minimum number needed.

- $Transitions$ defines the valid triplets (i, j, m) where system i can modernize by some means to system j in mission m.

- $MissionUpgrades$ defines the valid triplets (i, j, m) where system i can upgrade to system j in mission m. Here, the seed system i is consumed in the creation of system j. Note that $MissionUpgrades \subseteq Transitions$.

- $StorageUpgrades$ defines the valid pairs (i, j) where system i can upgrade to system j in storage. As in $MissionUpgrades$, the seed system i is consumed in the creation of j. Unlike $MissionUpgrades$, these upgrades happen in storage, not in the context of a mission, and so can define pairs (i, j) not seen in the set $Transitions$.

- $Inflow_{i,m}$ defines the set of valid systems j for which $(j, i, m) \in Transitions$. These are the systems that can modernize <u>to</u> system i in mission m.

- $Outflow_{i,m}$ defines the set of valid systems j for which $(i, j, m) \in Transitions$. These are the systems that can be modernized <u>from</u> system i in mission m.

- $PurchasableVeh$ defines the set of systems i that are able to be purchased. This is used to filter the $iNumBatchesPurchased_{i,t}$ variables down to the minimal number needed.

- $DeliverableVeh$ defines the set of systems i that are able to be delivered (i.e. finished production). This is used to filter the $bVehDelivered_{i,t}$ variables down to the minimal number needed.

- $FreeInterimUpgVeh$ defines the set of systems i that are 1) able to be upgraded to with zero cost and then 2) upgraded to something else. This is used to define the "hull" vehicles in storage.

Optimization Tiers

These tuples are used to pass optimal solutions from higher priority tiers on to the next tier.

- $fixedVehInMissionUpgraded$ defines the valid (i, j, m, t, N) quintuples where N vehicles will be upgraded from system i to j in mission m at time $t \leq T$ as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedVehFromStorage$ defines the valid (i, j, m, t, N) quintuples where N systems undergo a storage swap wherein system i leaves mission m and goes to storage while j leaves storage and enters mission m at time $t \leq T$ as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedModernizedDeficit$ defines the valid (i, m, t, N) quadruples where system i in mission m has a modernization deficit of N systems at time $t \leq T$ as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedFinalMandateDeficit$ defines the valid (i, m, N) triplets where system i in mission m has a final deficit of N systems at time T as determined by preceding tier optimizations. This set will be empty during the first tier.

Modernization Scheduling

- $MissionSuccessions$ defines the valid mission pairings (m_1, m_2) where mission m_1 must completely modernize and finish fielding before m_2 can begin fielding (m_1 is said to precede m_2).

- $PrecededMissions$ defines the set of missions m which are preceded by some other mission as defined in the $MissionSuccessions$ set.

- $SystemObviations$ defines the valid system pairings (i, j) where system j can only be produced before system i is produced (i obviates j).

- $SyncMissions_s$ defines the set of missions belonging to the synchronization set s.

- $SyncSystems_s$ defines the set of systems belonging to the synchronization set s.

- $UpgDensityFlags$ defines the triplets (i, m, ℓ) where system i must transition into mission m so as to either achieve one of up to 3 density levels $\ell \in UpgDensityLevels_m$ or exceed the highest density level. It helps filter the $bTransitionedToDensityLevel_{i,m,\ell}$ variables down to the minimal set needed.

- $FinalDensityFlags$ defines the triplets (i, m, ℓ) where system i in mission m must either achieve one of up to 3 density levels $\ell \in FinalDensityLevels_m$ or exceed the highest density level during the final time period. It helps filter the $bHasFinalDensity_{i,m,\ell}$ variables down to the minimal set needed.

- $StoragePriorityPairs$ defines the valid parings (i, j) where if system i is exchangeable in storage, then j cannot be spoken for as the seed of a storage upgrade.

- $UpgBeforePurch$ defines the set of systems i that must be used up as storage upgrade seeds before any corresponding new purchases can occur. That is, for all systems j such that (i, j) is a storage upgrade, no systems j may be purchased while i is available in storage.

Product Families

- $ProductFamily_p$ defines the set of system types belonging to the product family p.

- $VehPFMembership_i$ defines the set of product families to which system i belongs.

- $PFWithActive$ defines product families p for which a per-period active cost exists. This filters the $bProductFamilyActive_{p,t}$ variables down to the minimal number needed.

- $PFWithStartup$ defines product families p for which a start-up cost profile exists. This filters the $bProductFamilyStartup_{p,t}$ variables down to the minimal number needed.

- $PFWithProdCtrls$ defines product families p for which there exist certain controls on the production of member systems. These controls include a minimum production rate, a delivery variance limit for production smoothing, or the restriction that production gaps are not allowed. This filters the $bProductFamilyDelivered_{p,t}$ variables down to the minimal number needed.

- $PFWithLrip$ defines product families p which has an LRIP profile. This filters the $bProductFamilyFrpStarted_{p,t}$ variables down to the minimal number needed.

- $LripProfiles$ defines the valid pairs (p, i) where system i of product family p has an LRIP profile.

- $LripYears$ denotes the set of integers $\{1, 2, \ldots, maxLripYear\}$, where $maxLripYear$ gives the maximum number of years prior to full-rate production, across all LRIP profiles, that LRIP systems are produced.

- $PFWithRdte$ defines product families p for which at least one RDT&E cost profile exists. This filters the $bRdteDelay_{p,d}$ variables down to the minimial number needed.

- $AllowedDelays_p$ defines the delays d for which a product family RDT&E cost profile exists. These are the valid time periods by which the effort may be delayed, given that delays are allowed.

Future Programs

- $FutureProgram_{\mathcal{F}}$ defines the set of future system types belonging to program $\mathcal{F}$.

- $FutureTransitions$ defines the valid triplets $(i, \mathcal{J}, m)$ where system i can be replaced by future system $\mathcal{J}$ in mission m.

- $FutureTransitionPurchases$ is a subset of $FutureTransitions$ and defines the valid transition $(i, \mathcal{J}, m)$ where system i can be replaced by future system $\mathcal{J}$ in mission m and system i returns to storage.

- $FutureMissionMap_{\mathcal{J}}$ defines the mission to which future system $\mathcal{J}$ is fielded.

- $FutureMandatedSystems$ defines the set of future system types that must field.

- $FutureUpgDensityFlags$ defines triplets $(\mathcal{J}, m, \ell)$ where system $\mathcal{J}$ must transition into mission m to either achieve one of up to 3 density levels $\ell \in UpgDensityLevels_m$ or exceed the highest density level - filtering $bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell}$ down to the minimal set needed.

- $FutureFinalDensityFlags$ defines the triplets $(\mathcal{J}, m, \ell)$ where future system $\mathcal{J}$ in mission m must either achieve one of up to 3 density levels $\ell \in FinalDensityLevels_m$ or exceed the highest density level during the final time period. This filters the $bFutureHasFinalDensity_{\mathcal{J},m,\ell}$ variables down to the minimal set needed.

This page intentionally left blank.

Chapter 3

Model Input Parameters

The parameters below capture the specific qualities of the business rules as they relate to a broad range of performance, fielding, and budgetary requirements.

Optimization Tiers and Phases

- $\alpha_{i,m}$ gives the performance of system type i in mission m. This is the key parameter driving the performance optimization phase, which seeks to maximize the sum of performance for all system types for all missions over the entire planning horizon.
- $CurrentTier$ gives the tier number for the missions that are currently being modernized. Missions with a smaller tier number have already be modernized and their schedule is fixed. Missions with a larger tier number have not yet been modernized, and their schedule (of no modernizations) is also fixed.
- $Tier_m$ gives the priority tier for mission m. All missions with the smallest tier number are modernized together first, then missions with the next smallest tier number are modernized with the budget left over from the first tier. This continues in the same manner until all tiers are modernized.
- $SchedulePhase$, $BudgetPhase$, $PerformancePhase$, and $CostPhase$ are binary parameters indicating which of the four phases the optimization is currently performing. They are used in the objective function to ensure that the proper terms are being minimized or maximized.
- $TierScheduleDeficitBound$ gives the upper limit on the schedule violation amount allowed during phases subsequent to the schedule phase within a tier. This limit equals the minimum schedule violation discovered during the schedule phase.
- $BudgetOverrunBound$ gives the upper limit on the budget overage amount allowed during phases subsequent to the budget phase. This limit equals the minimum budget overrun amount discovered in the budget phase.
- $MinimumPerformance$ gives the smallest acceptable value for total fleet performance during phases subsequent to the performance phase. This lower bound equals the maximum cumulative fleet performance found during the performance phase.

Fleet Structure and Flow

- $PurchBatchSize_i$ gives the smallest number of systems i that can be purchased. Furthermore, all purchases must be made in multiples of this batch size. For example, if $PurchBatchSize_i = 5$, then the number of i that can be purchased at any time is 0, 5, 10, 15, 20, etc.

- $AllowPrePurchasing$ is a binary flag that is set to 1 if pre-purchasing is allowed and 0 if it is not. When pre-purchasing is allowed, systems purchased or upgraded in storage are not required to be fielded into a mission as soon as their production is complete. When pre-purchasing is disallowed, these purchased and in-storage-upgraded systems must be immediately fielded to a mission.

- $PurchDelay_i$ gives the number of time periods between when costs are incurred and when the system i is actually delivered. $PurchDelay_i$ is itself the sum of two separate delay parameters $PurchAdminDelay_i$ followed by $PurchProdDelay_i$.

- $UpgDelay_{i,j}$ gives the number of time periods between when costs are incurred for upgrading system i to j and when the resultant j is actually delivered. This parameter applies to both in-mission and in-storage upgrades. $UpgDelay_{i,j}$ is the sum of parameters $UpgAdminDelay_{i,j}$ followed by $UpgProdDelay_{i,j}$.

- $LripDelay_{p,i}$ gives the production delay when obtaining LRIP systems i for family p. Depending on whether the LRIP systems are purchased or upgraded from a seed, $LripDelay_{p,i}$ is equal to $PurchProdDelay_i$ or $UpgProdDelay_{j,i}$, respectively, where j is the seed system.

- $VehPerMission_m$ gives the total number of systems needed for mission m. Note that this value must equal the total number of systems in the initial inventory for m.

- $BrigPerMission_m$ gives the total number of groups needed for mission m.

- $VehPerBrigade_m$ gives the number of systems per group for mission m. This is used to ensure that systems are upgraded in group-sized increments.

- $InitialVehInMission_{i,m}$ gives the initial number of systems i in mission m. This parameter describes the complete existing fleet prior to any modernization. Note that the total initial inventory for each mission must match the number of systems required for that mission given by the $VehPerMission_m$ parameter.

- $InitialBrigInMission_{i,m}$ gives the initial number of groups of system i in mission m. This is the same information as given in $InitialVehInMission$, just denominated in group size.

- $InitialVehInStorage_i$ gives the initial number of systems i in storage at the beginning of the planning horizon.

- $maxPathLength_m$ gives the maximum possible path length in the transition diagram of mission m.

Modernization Scheduling

- $ModernPercent_{i,m,t}$ gives the minimum percentage of system i in mission m that must be modernized to something else by time $t \leq T$. Note that system i must be in the initial inventory for m.

- $FinalMandate_{i,m}$ gives the minimum number of systems i in mission m that must be in service at the end of the conventional time horizon (T). This is often used when administrative requirements force certain programs to be mandatory.

- $YearlyBrigTransitLimit_m$ gives the maximum number of groups that can be transitioned in mission m during a single time period. This is used since groups are not available for modernization all at once due, for example, to deployment based on the ARFORGEN cycle.

- $CumulativeBrigTransitLimit_m$ gives the maximum number of groups for mission m that can be transitioned throughout the conventional planning horizon.

- $UpgDensityLevels_m$ defines the transition density levels for mission m. There may be anywhere between 0 and 3 levels defined. As an example, if $UpgDensityLevels_m = \{10, 13, 20\}$, then the number of groups transitioned into m of a specific system type must be either 10, 13, 20, or greater than 20.

- $FinalDensityLevels_m$ defines the density levels for mission m at the end of the conventional planning horizon. There may be anywhere between 0 and 3 levels defined. Consider the case where $FinalDensityLevels_m = \{10, 13, 20\}$. Then the number of groups in m during time T of a specific system type must be either 10, 13, or at least 20.

- $ProdFamilyMaxDelivery_{p,t}$ gives the upper limit on the number of systems from product family p that can be delivered at time $t \leq T$.

- $ProdFamilyMinDelivery_p$ gives the lower limit on the number of systems from product family p that must be delivered at any time period, given that systems from the family are delivered.

- $ProdFamilyMaxCumulativeDelivery_p$ gives the upper limit on the cumulative number of systems from product family p that can ever be delivered.

- $ProdFamilyAllowGaps_p$ is a binary flag that takes on value 1 to indicate that production gaps for family p are allowed.

- $MaxDeliveryVariance_p$ gives the bandwidth within which the product family p must stay after it reaches full-rate production (that is, after LRIP and/or ramp-up periods). For example, if $fMedianDeliveryLevel_p = 100$ and $MaxDeliveryVariance_p = 0.2$, then the highest number of systems delivered from family p in a time period can only be $1.1 * 100$ and the lowest number of systems delivered can only be $0.9 * 100$.

- $RampUp_p$ gives the number of time-periods prior to full-rate production that the family p is ramping-up the production line. These periods do not count towards to $MaxDeliveryVariance_p$ bandwidth. The only restriction is that each successive ramp-up period must produce at least as many systems as the previous.

- $LripPreProduction_{p,i,t}$ gives the number of systems i of product family p that are produced t years before full-rate production.

- $LripPreDelivery_{p,i,t}$ gives the number of systems i of product family p that are produced t years before full-rate production and are then delivered to storage.

- $LripSeed_{p,i}$ gives the seed system (if any) that is used in producing LRIP systems i for product family p. This helps determine what LRIP production costs and delays are incurred: if no seed is given then the purchase cost/delays for system i are used, if the seed (say system j) is given then the corresponding upgrade from j to i costs/delays are used.

- $LripConsumesSeeds_{p,i}$ is a binary flag indicating whether the seed system (given by $LripSeed_{p,i}$) is removed from storage in producing LRIP systems i in product family p.

Cost & Budgets

- $ProdFamilyStartupCostSched_{p,t}$ gives the start-up cost schedule for product family p during its t^{th} time period of activity. Note the $t = 0$ represents the time when the family first becomes active, $t = 1$ represents the year after the family first becomes active, and $t = -1$ represents the year before the family first becomes active. All are valid time periods to incur start-up costs.

- $ProdFamilyActiveCost_p$ gives the recurring cost of keeping product family p active for one time period.

- $PrePurchCost_i + PurchCost_i$ gives the cost of purchasing 1 system of type i. Here, $PurchCost_i$ gives the amount charged in the time period work first begins to deliver the system (usually the admin period). If a purchase requires long lead, then $PrePurchCost_i$ gives the amount charged one period prior to $PurchCost_i$; otherwise $PrePurchCost_i = 0$.

- $PreUpgCost_{i,j} + UpgCost_{i,j}$ gives the cost of upgrading (either in mission or in storage) 1 system from type i to type j. $UpgCost_{i,j}$ gives the amount charged in the time period work first begins to deliver the system. If an upgrade requires long lead, then $PreUpgCost_{i,j}$ gives the amount charged one period prior to $UpgCost_{i,j}$; otherwise $PreUpgCost_{i,j} = 0$.

- $LripPreCost_{p,i} + LripCost_{p,i}$ denotes the cost of procuring 1 LRIP system of type i in product family p. Here, $LripCost_{p,i}$ gives the amount charged in the time period work first begins to deliver the LRIP system (usually the admin period). If a procurement requires long lead, then $PreLripCost_{p,i}$ gives the amount charged one period prior to $LripCost_{p,i}$; otherwise $PreLripCost_{p,i} = 0$.

- $VehOSCost_{i,m}$ gives the cost for operating and supporting system i in mission m for one time period.

- $RdteCost_{p,d,t}$ gives the RDT&E cost for product family p at time $t \leq T$ when it is delayed by d time periods. This entire cost profile must be incurred in order for any systems in product family p to be fielded.

- $ProcureBudget_t$ gives the per-period budget at time $t \leq T$ for procurement of systems. Procurement expenses include system purchase and upgrade costs (both LRIP and FRP), as well as product family active and start-up costs. When procurement expenses for future programs are incurred during the conventional time horizon, those expenses must also respect this budget.

- $OSBudget_t$ gives the per-period budget at time $t \leq T$ for O&S expenditures for systems in the fleet. Systems in storage do not incur O&S expenses. When O&S expenses for future programs are incurred during the conventional time horizon, those expenses must also respect this budget.

- $RdteBudget_t$ gives the per-period budget at time $t \leq T$ for expenditures on RDT&E efforts. When RDT&E expenses for future programs are incurred during the conventional time horizon, those expenses must also respect this budget.

- $CombinedBudget_t$ gives the budget at time t for a user-specified combination of procurement, O&S, and RDT&E expenditures. For example, the user may constrain all three categories under this budget; or perhaps only procurement and RDT&E expenditures are desired to be constrained together. When expenses for future programs are incurred during the conventional time horizon, those expenses must also respect this budget.

- $CumulativeProcureBudget$ gives the cumulative budget for all procurement expenses ever incurred, both over the conventional and extended planning horizons.

- $CumulativeOSBudget$ gives the cumulative budget for O&S expenses ever incurred, both over the conventional and extended planning horizons.

- $CumulativeRdteBudget$ gives the cumulative budget for RDT&E expenses ever incurred, both over the conventional and extended planning horizons.

- $CumulativeCombinedBudget$ gives the cumulative budget for all (user-specified) combined expenses ever incurred, both over the conventional and extended planning horizons.

Future Programs

- $FutureFieldingProfile_{\mathcal{J},t}$ gives the number of groups of future systems of type $\mathcal{J}$ that must be fielded at time $t \leq \mathcal{T}$, if the future system is activated.

- $FutureSysFirstField_{\mathcal{J}}$ gives the time period during which future system $\mathcal{J}$ first fields, according to the schedule dictated by $FutureFieldingProfile_{\mathcal{J},t}$.

- $FutureProcureBudget_t$, $FutureOSBudget_t$, $FutureRdteBudget_t$, and $FutureCombinedBudget_t$ give the procurement, O&S, RDT&E, and Combined budgets, respectively, for time periods in the extended time horizon (i.e. $T < t \leq \mathcal{T}$).

- $FutureRdteCostSchedule_{\mathcal{F},t}$, $FutureStartupCostSchedule_{\mathcal{F},t}$, and $FutureActiveCostSchedule_{\mathcal{F},t}$ give the cost of future program $\mathcal{F}$ (if activated) at time t in the categories of RDT&E, startup, and active cost, respectively.

- $FutureTransitCost_{i,\mathcal{J},m}$ gives the cost of transitioning 1 system from type i to future system type $\mathcal{J}$ in mission m.

- $FutureVehOSCost_{\mathcal{J}}$ gives the cost of operating and supporting future system $\mathcal{J}$ for one time period.

- $FutureLongLead_{i,\mathcal{J},m}$ gives the fraction of the transition cost from system i to future system $\mathcal{J}$ in mission m that is incurred a year earlier than normal.

- $FutureDelay_{i,\mathcal{J},m}$ gives the number of time periods between when transition costs are incurred and when the system i is replaced by $\mathcal{J}$ in mission m.

- $FutureLripProfile_{\mathcal{F},t}$ gives the number of LRIP systems that program $\mathcal{F}$ produces at time t.

- $FutureLripDelay_{\mathcal{J}} = FutureDelay_{i,\mathcal{J},m}$ for that triplet $(i, \mathcal{J}, m)$ having the largest $FutureTransitCost_{i,\mathcal{J},m}$.

- $FutureLripLongLead_{\mathcal{J}} = FutureLongLead_{i,\mathcal{J},m}$ for that triplet $(i, \mathcal{J}, m)$ having the largest $FutureTransitCost_{i,\mathcal{J},m}$.

- $FutureLripCost_{\mathcal{J}} = FutureTransitCost_{i,\mathcal{J},m}$ for that triplet $(i, \mathcal{J}, m)$ having the largest $FutureTransitCost_{i,\mathcal{J},m}$.

Auxiliary Parameters

These parameters are not input by the user, but are instead calculated from the value of other user-entered parameters.

- $FirstAvailable_i$ gives the time period at which system type i first becomes available for delivery. If the system is never available, then the parameter has value $T + 1$.

- $MaxYearlyBrigTransitLimit$ gives $\max_m YearlyBrigTransitLimit_m$.
- $TotalVehPopulation$ gives the total number of systems in the fleet.
- $NumFutureSystems$ is the total number of <u>types</u> of future systems.

This page intentionally left blank.

Chapter 4

MILP Decision Variables

The following is a list of all decision variables used in the MILP formulation. Notice that the integer $iModernizedDeficit_{i,m,t}$ and $iFinalMandateDeficit_{i,m}$ variables along with the continuous "Budget Overrun" variables are used to diagnose business rule violations where the optimization plan is unable to meet strict user-specified upgrade mandates or budgetary limits.

Non-Negative Integer Variables

- $iNumBrigInMissionUpgraded_{i,j,m,t}$ denotes the number of groups of system type i which are upgraded to system type j in mission m at time t. It is defined for all $(i, j, m) \in MissionUpgrades$ and for all $t \leq T$.

- $iNumBrigFromStorage_{i,j,m,t}$ denotes the number of groups of system i that are sent out of mission m to storage in exchange for system j from storage into mission m at time t. This is defined for all $(i, j, m) \in Transitions$ and for all $t \leq T$.

- $iNumBatchesPurchased_{i,t}$ denotes the number of batches (whose size is given by $PurchBatchSize_i$) of system i purchased at time t. These purchased systems enter storage and may immediately be sent out to a mission. This is defined for all $i \in PurchasableVeh$ and for all $t \leq T$.

- $iNumInStorageUpgraded_{i,j,t}$ denotes the number of systems of type i upgraded in storage to type j at time t. This is defined for all $(i, j) \in StorageUpgrades$ and for all $t \leq T$.

- $iModernizedDeficit_{i,m,t}$ denotes how far system i in mission m is below the system modernization requirement (given by $ModernPercent_{i,m,t}$) at time t . Any positive value indicates a business rule violation. This variable is defined for all $(i, m) \in Roles$ and for all $t \leq \mathcal{T}$.

- $iFinalMandateDeficit_{i,m}$ denotes how far system i in mission m is below the end-of-conventional-horizon system mandate (given by $FinalMandate_{i,m}$). This variable is denominated by group size (so if a mission is 17 systems below mandate and the group size is 10, then this variable reports 2 brigades under mandate). Note that the output reports the correct deficit in number of systems. Any positive value indicates a business rule violation. This variable is defined for all $(i, m) \in MandatedRoles$.

- $iNumBrigReplaced_{i,\mathcal{J},m,t}$ denotes how many groups of system i are replaced by future system $\mathcal{J}$ in mission m at time t. This variable is defined for all $(i, \mathcal{J}, m) \in FutureTransitions$ and for all $t \leq \mathcal{T}$.

- $iFutureSystemMandateDeficit_{\mathcal{J}}$ denotes how many groups of future system $\mathcal{J}$ are not fielded in the conventional or extended time horizons.

Binary Variables

- $bTransitionedToDensityLevel_{i,m,\ell}$ denotes if mission m ever has system i at one of the three density levels $\ell \in UpgDensityLevels_m$. This is defined for all $(i, m, \ell) \in UpgDensityFlags$.

- $bHasFinalDensity_{i,m,\ell}$ denotes whether system i in mission m during time $\mathcal{T}$ achieves one of the three density levels $\ell \in FinalDensityLevels_m$. This is defined for all $(i, m, \ell) \in FinalDensityFlags$.

- $bVehDelivered_{i,t}$ denotes whether at least 1 system of type i is delivered (i.e. completes production) at time t. This is defined for all $i \in DeliverableVeh$ and for all $t \leq T$.

- $bVehInStorageExchangeable_{i,t}$ indicates whether there is at least 1 system of type i that is exchangeable in storage at time t. It is defined for all $i \in PrecedingSystems$ and for all $t \leq T$.

- $bUpgBeforePurchInStorageExchangeable_{i,t}$ indicates whether there is at least 1 system of type i that is exchangeable in storage at time t. It is defined for all $i \in UpgBeforePurch$ and for all $t \leq T$.

- $bLripVehBaseYear_{p,i,t}$ denotes whether system i of product family p first delivers non-LRIP assets (i.e. FRP) at time t. It is defined for all $(p, i) \in LripProfiles$ and for all t.

- $bProductFamilyActive_{p,t}$ denotes whether product family p is active at time t. This is needed only for families having a per-period active cost. Hence, it is defined for all $p \in PFWithActive$ and for all $t \leq T$.

- $bProductFamilyStartup_{p,t}$ denotes whether product family p first becomes active at time $t \leq T$.

- $bMissionCanField_{m,t}$ denotes whether mission m is allowed to field in time t based on the completion of predecessor missions. Since it is only needed for missions that are preceded by some other mission, it is defined for all $m \in PrecededMissions$ and for all $t \leq T$.

- $bInterimBrigCanField_{i,m,t}$ denotes whether intermediate systems i in mission m are allowed (but not required) to field in time period t. This is defined for all $(i, m) \in InterimRoles$ and for all $t \leq T$.

- $bFutureProrgam_{\mathcal{F}}$ denotes whether future program $\mathcal{F}$ is activated.

- $bFutureSystem_{\mathcal{J}}$ denotes whether future system $\mathcal{J}$ is activated.

- $bFutureSystemDeficit_{\mathcal{J}}$ denotes that a mandated future system $\mathcal{J}$ does not field.

- $bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell}$ denotes if mission m ever has system $\mathcal{J}$ at one of the three density levels $\ell \in UpgDensityLevels_m$. This is defined for all $(\mathcal{J}, m, \ell) \in FutureUpgDensityFlags$.

- $bFutureHasFinalDensity_{\mathcal{J},m,\ell}$ denotes whether future system $\mathcal{J}$ in mission m during time $\mathcal{T}$ achieves one of the three density levels $\ell \in FinalDensityLevels_m$. This is defined for all $(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$.

Continuous "Binary" Variables

These variables are continuous in the range $[0, 1]$, but are explicitly restricted to binary values by the nature their associated constraints.

- $bVehEverDelivered_i$ denotes whether at least 1 system of type i is ever delivered. This is defined for all $i \in DeliverableVeh$.

- $bProductFamilyDelivered_{p,t}$ denotes whether any system from family p has been delivered at time $t \leq T$.

- $bProductFamilyFrpStarted_{p,t}$ denotes whether production in family p starts delivering FRP assets at time $t \leq T$.

- $bRdteDelay_{p,d}$ denotes whether product family p is delayed by d time periods. If $bRdteDelay_{p,d} = 1$, then the associated cost profile $RdteCost_{p,d,t}$ is used. If product family p is not used, then $bRdteDelay_{p,d} = 0$ for all d.

Non-negative Continuous Variables

The first set of variables is used to represent budget overages in the categories of Procurement, O&S, and RDT&E, as well as a user-specified combination of the three. The first four sets of variables denote per-period overages in these categories during the conventional time horizon. The next four denote the per-period overages during the extended time horizon. The last four denote budget overages for the entire planning horizon (conventional plus extended horizons). These are used to help diagnose modernization plans whose constraints cannot be met without going over budget.

- $fProcureBudgetOverrun_t$ for all $t \leq T$
- $fOSBudgetOverrun_t$ for all $t \leq T$
- $fRdteBudgetOverrun_t$ for all $t \leq T$
- $fCombinedBudgetOverrun_t$ for all $t \leq T$
- $fFutureProcureBudgetOverrun_t$ for all $T < t \leq \mathcal{T}$
- $fFutureOSBudgetOverrun_t$ for all $T < t \leq \mathcal{T}$
- $fFutureRdteBudgetOverrun_t$ for all $T < t \leq \mathcal{T}$
- $fFutureCombinedBudgetOverrun_t$ for all $T < t \leq \mathcal{T}$
- $fCumulativeProcureBudgetOverrun$
- $fCumulativeOSBudgetOverrun$
- $fCumulativeRdteBudgetOverrun$
- $fCumulativeCombinedBudgetOverrun$

The following variable is used to represent the median product family production level for each individual product family and is only used for the **Production Smoothing** business rule

- $fMedianDeliveryLevel_p$

Chapter 5

MILP Variable Expressions

The following variable expressions are used to conveniently capture additional information about the model. These expressions are defined as fixed linear functions of input parameters and decision variables. They greatly aid readability of the formulation without adding the computational complexity of new variables. Note that the optimization code contains approximately 15 additional variable expressions than the ones documented below. However, these addition expressions serve an auxiliary role (mainly for aiding output of optimization data) and do not affect the mathematical structure in the formulation itself. For that reason, we have opted not to include these auxiliary expressions in this document.

Fleet Structure and Flow

- $NumVehInMissionUpgraded_{i,j,m,t}$ denotes the number of systems i transitioned by an in-mission upgraded to system j for mission m at time t. Note that this is just a redenomination of the variable $iNumBrigInMissionUpgraded_{i,j,m,t}$.

$$\forall (i,j,m) \in MissionUpgrades, t \leq T$$
$$NumVehInMissionUpgraded_{i,j,m,t} = iNumBrigInMissionUpgraded_{i,j,m,t} * VehPerBrigade_m \quad (5.1)$$

- $NumVehFromStorage_{i,j,m,t}$ denotes the number of systems i that are swapped out for system j in mission m at time t. Here, i is sent to storage while j is pulled from storage. Note that this is just a redenomination of the variable $iNumBrigFromStorage_{i,j,m,t}$.

$$\forall (i,j,m) \in Transitions, t \leq T$$
$$NumVehFromStorage_{i,j,m,t} = iNumBrigFromStorage_{i,j,m,t} * VehPerBrigade_m \quad (5.2)$$

- $NumBrigInMission_{i,m,t}$ denotes the number of groups of system i performing in mission m at time t.

$$\forall (i,m) \in Roles, t \leq \mathcal{T}$$

$$\begin{aligned}
NumBrigInMission_{i,m,t} &= InitialBrigInMission_{i,m} \\
&+ \sum_{\substack{j,t^*: \\ j \in Inflow_{i,m} \\ t^* \leq \min\{t,T\}}} iNumBrigFromStorage_{j,i,m,t^*} \\
&+ \sum_{\substack{j,t^*: \\ (j,i,m) \in MissionUpgrades \\ t^* \leq \min\{t,T\}}} iNumBrigInMissionUpgraded_{j,i,m,t^*} \\
&- \sum_{\substack{j,t^*: \\ j \in Outflow_{i,m} \\ t^* \leq \min\{t,T\}}} iNumBrigFromStorage_{i,j,m,t^*} \\
&- \sum_{\substack{j,t^*: \\ (i,j,m) \in MissionUpgrades \\ t^* \leq \min\{t,T\}}} iNumBrigInMissionUpgraded_{i,j,m,t^*} \\
&- \sum_{\substack{\mathcal{J},t^*: \\ (i,\mathcal{J},m) \in FutureTransitions \\ t^* \leq t}} iNumBrigReplaced_{i,\mathcal{J},m,t^*}
\end{aligned} \tag{5.3}$$

- $NumVehInMission_{i,m,t}$ denotes the number of systems i performing in mission m at time t.

$$\forall (i,m) \in Roles, t \leq \mathcal{T}$$

$$NumVehInMission_{i,m,t} = NumBrigInMission_{i,m,t} * VehPerBrigade_m \tag{5.4}$$

- $NumVehMissionExchangeable_{i,m,t}$ denotes the number of systems i in mission m that are <u>not</u> "spoken for" (i.e. in production for a future mission upgrade) in time t.

$$\forall (i,m) \in Roles, t \leq T$$

$$\begin{aligned}
&NumVehMissionExchangeable_{i,m,t} = \\
&\quad NumVehInMission_{i,m,t} \\
&- \sum_{j:\ (j,i,m) \in MissionUpgrades} NumVehInMissionUpgraded_{j,i,m,t} \\
&- \sum_{j \in Inflow_{i,m}} NumVehFromStorage_{j,i,m,t} \\
&- \sum_{\substack{j,t^*: \\ (i,j,m) \in MissionUpgrades \\ t < t^* \leq t + UpgProdDelay_{i,j}}} NumVehInMissionUpgraded_{i,j,m,t^*}
\end{aligned} \tag{5.5}$$

- $TransitionedToRole_{i,m}$ denotes whether system i ever fields to mission m. Caution, if there is neither upgrade density nor final density requirements for mission m, then this expression will take on value 0 regardless of whether systems i transitioned into m. This is currently acceptable since this expression is only used in cases where there exists a final density requirement.

$$\begin{aligned}&\forall (i,m) \in Roles\\ &TransitionedToRole_{i,m} = \sum_{\ell:\ (i,m,\ell)\in UpgDensityFlags} bTransitionedToDensityLevel_{i,m,\ell}\end{aligned} \tag{5.6}$$

- $NumBrigTransit_{i,j,m,t}$ denotes the number of systems i transitioned to system j in mission m at time t. Recall that "transition" refers to both the mission upgrades and storage swaps.

$$\begin{aligned}&\forall (i,j,m) \in Transitions, t \leq T\\ &\qquad NumBrigTransit_{i,j,m,t} = iNumBrigFromStorage_{i,j,m,t}\\ &\qquad\qquad + iNumBrigInMissionUpgraded_{i,j,m,t}\end{aligned} \tag{5.7}$$

- $CumulativeBrigRetiredFromRole_{i,m,t}$ denotes the total number of groups of systems i that are transitioned out of mission m during time periods up to and including t.

$$\begin{aligned}&\forall (i,m) \in Roles, t \leq \mathcal{T}\\ &CumulativeBrigRetiredFromRole_{i,m,t} = \sum_{\substack{j,t^*:\\ j\in Outflow_{i,m}\\ t^*\leq \min\{t,T\}}} iNumBrigFromStorage_{i,j,m,t^*}\\ &\qquad + \sum_{\substack{j,t^*:\\ (i,j,m)\in MissionUpgrades\\ t^*\leq \min\{t,T\}}} iNumBrigInMissionUpgraded_{i,j,m,t^*}\\ &\qquad\qquad + \sum_{\substack{\mathcal{J},t^*\\ (i,\mathcal{J},m)\in FutureTransitions\\ t^*\leq t}} iNumBrigReplaced_{i,\mathcal{J},m,t^*}\end{aligned} \tag{5.8}$$

- $NumVehReplaced_{i,\mathcal{J},m,t}$ denotes how many systems i are replaced by future system $\mathcal{J}$ in mission m at time t. This variable is defined for all $(i,\mathcal{J},m) \in FutureTransitions$ and for all $t \leq \mathcal{T}$.

$$\begin{aligned}&\forall (i,\mathcal{J},m) \in FutureTransitions, t \leq \mathcal{T}\\ &\qquad NumVehReplaced_{i,\mathcal{J},m,t} = iNumBrigReplaced_{i,\mathcal{J},m,t} * VehPerBrigade_m\end{aligned} \tag{5.9}$$

Low-Rate Initial Production

- $LripDelivered_{p,i,t}$ denotes the number of LRIP systems i in product family p that are delivered to storage at time period t.

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$LripDelivered_{p,i,t} = \sum_{\substack{t^* \in LripYears \\ t+t^* \leq T}} LripPreDelivery_{p,i,t^*} * bLripVehBaseYear_{p,i,t+t^*} \quad (5.10)$$

- $LripSeedsConsumed_{i,t}$ denotes the number of systems i that are consumed in the production of LRIP systems and removed from storage in time period t. Note that LRIP seeds are consumed (removed from storage) during the first production period for that upgrade.

$$\forall i, t \leq T$$
$$LripSeedsConsumed_{i,t} = \sum_{\substack{p,j,t^*: \\ (p,j) \in LripProfiles \\ t^* \in LripYears \\ LripSeed_{p,j}=i \\ LripConsumesSeeds_{p,j}=1 \\ t+t^*+LripDelay_{p,j} \leq T}} LripPreProduction_{p,j,t^*} * bLripVehBaseYear_{p,j,t+t^*+LripDelay_{p,j}} \quad (5.11)$$

- $NumLripVehActive_{i,t}$ denotes the number of systems i that are in administration or production periods (i.e. active) <u>due to LRIP</u> at time period t. Here, $\beta_{p,i}$ is a binary paramater that takes value 1 if $LripDelay_{p,i} = 0$ and 0 if $LripDelay_{p,i} > 0$.

$$\forall i, t \leq T$$
$$NumLripVehActive_{i,t} = \sum_{\substack{p,t^*,t^{**}: \\ p \in VehPFMembership_i \\ t^* \in LripYears \\ t^{**}-t^*-LripDelay_{p,i} \leq t < t^{**}-t^*+\beta_{p,i}}} LripPreProduction_{p,i,t^*} * bLripVehBaseYear_{p,i,t^{**}} \quad (5.12)$$

Storage

- $NumVehPurchased_{i,t}$ denotes the number of systems i purchased at time t. These purchases are placed directly into storage.

$$\forall i \in PurchasableVeh, t \leq T$$
$$NumVehPurchased_{i,t} = PurchBatchSize_i * iNumBatchesPurchased_{i,t} \quad (5.13)$$

- $NumVehInStorage_{i,t}$ denotes the number of systems i that are currently in storage at time t. This is calculated by adding what you start within in storage, plus what flows into storage up to time t, minus what flows out of storage up to time t. Hence this expression also counts systems that are "spoken for" in that they are in the middle of production for a future in-storage upgrade.

$$\forall i, t \leq T$$

$$\begin{aligned}
NumVehInStorage_{i,t} = {} & InitialVehInStorage_i \\
& + \sum_{t^* \leq t} NumVehPurchased_{i,t^*} \\
& + \sum_{\substack{j,m,t^*: \\ (i,j,m) \in Transitions \\ t^* \leq t}} NumVehFromStorage_{i,j,m,t^*} \\
& + \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpgrades \\ t^* \leq t}} iNumInStorageUpgraded_{j,i,t^*} \\
& + \sum_{\substack{p,t^*: \\ (p,i) \in LripProfiles \\ t^* \leq t}} LripDelivered_{p,i,t^*} \\
& - \sum_{\substack{j,m,t^*: \\ (j,i,m) \in Transitions \\ t^* \leq t}} NumVehFromStorage_{j,i,m,t^*} \\
& - \sum_{\substack{j,t^*: \\ (i,j) \in StorageUpgrades \\ t^* \leq t + UpgDelay_{i,j}}} iNumInStorageUpgraded_{i,j,t^*} \\
& - \sum_{t^* \leq t} LripSeedsConsumed_{i,t^*} \\
& + \sum_{\substack{\mathcal{J},m,t^*: \\ (i,\mathcal{J},m) \in FutureTransitionPurchases \\ t^* \leq t}} NumVehReplaced_{i,\mathcal{J},m,t^*}
\end{aligned} \tag{5.14}$$

- $NumVehInStorageExchangeable_{i,t}$ denotes the number of systems i in time period t thate are <u>not</u> "spoken for" by a future in-storage upgrade.

$$\forall i, t \leq T$$

$$\begin{aligned}
NumVehInStorageExchangeable_{i,t} = {} & NumVehInStorage_{i,t} \\
& - \sum_{\substack{j,t^*: \\ (i,j) \in StorageUpgrades \\ t < t^* \leq t + UpgProdDelay_{i,j}}} iNumInStorageUpgraded_{i,j,t^*}
\end{aligned} \tag{5.15}$$

Cost and Budgets

- $ProdFamilyStartupCost_{p,t}$ gives the startup expense incurred at time t for family p.

$$\forall p \in PFWithStartup, t \leq T$$

$$ProdFamilyStartupCost_{p,t} = \sum_{t^* \leq T} bProductFamilyStartup_{p,t^*} * ProdFamilyStartupCostSched_{p,t-t^*} \quad (5.16)$$

- $ProcureExpense_t$ denotes the amount spent on procurement of non-future systems (in storage upgrades, in mission upgrades, purchases, product families, and LRIP) at time t within the conventional time horizon.

$$\forall t \leq T$$

$$\begin{aligned} ProcureExpense_t = & \sum_{i \in PurchasableVeh} NumVehPurchased_{i,t+PurchDelay_i+1} * PrePurchaseCost_i \\ & + \sum_{i \in PurchasableVeh} NumVehPurchased_{i,t+PurchDelay_i} * PurchCost_i \\ & + \sum_{(i,j,m) \in MissionUpgrades} NumVehInMissionUpgraded_{i,j,m,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\ & + \sum_{(i,j,m) \in MissionUpgrades} NumVehInMissionUpgraded_{i,j,m,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\ & + \sum_{(i,j) \in StorageUpgrades} iNumInStorageUpgraded_{i,j,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\ & + \sum_{(i,j) \in StorageUpgrades} iNumInStorageUpgraded_{i,j,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\ & + \sum_{p \in PFWithActive} bProductFamilyActive_{p,t} * ProdFamilyActiveCost_p \\ & + \sum_{p} ProdFamilyStartupCost_{p,t} \\ & + \sum_{\substack{p,i,t^*: \\ (p,i) \in LripProfiles \\ t+t^*+LripDelay_{p,i}+1 \leq T}} \begin{pmatrix} LripPreProduction_{p,i,t^*} * LripPreCost_{p,i} \\ *bLripVehBaseYear_{p,i,t+t^*+LripDelay_{p,i}+1} \end{pmatrix} \\ & + \sum_{\substack{p,i,t^*: \\ (p,i) \in LripProfiles \\ t+t^*+LripDelay_{p,i} \leq T}} \begin{pmatrix} LripPreProduction_{p,i,t^*} * LripCost_{p,i} \\ *bLripVehBaseYear_{p,i,t+t^*+LripDelay_{p,i}} \end{pmatrix} \end{aligned} \quad (5.17)$$

- $FutureProcureExpense_t$ denotes the total amount spent on procurement of future systems (i.e. future system transition costs for LRIP and non-LRIP along with future program active and startup costs) at time t within the conventional and extended time horizons. Here the binary parameter $\gamma_{i,\mathcal{J},k}$ equals 1 if $FutureLongLead_{i,\mathcal{J},m} > 0$ and 0 otherwise. Also, the binary parameter $\eta_{\mathcal{J}}$ equals 1 if $FutureLripLongLead_{\mathcal{J}} > 0$ and 0 otherwise.

$$\forall t \leq \mathcal{T}$$

$$\begin{aligned}
&FutureProcureExpense_t = \\
&\sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m)\in FutureTransitions \\ t^*=t+FutureDelay_{i,\mathcal{J},m}+\gamma_{i,\mathcal{J},m}\leq \mathcal{T}}} \begin{pmatrix} NumVehReplaced_{i,\mathcal{J},m,t^*} * FutureTransitCost_{i,\mathcal{J},m} \\ *FutureLongLead_{i,\mathcal{J},m} \end{pmatrix} \\
&+ \sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m)\in FutureTransitions \\ t^*=t+FutureDelay_{i,\mathcal{J},m}\leq \mathcal{T}}} \begin{pmatrix} NumVehReplaced_{i,\mathcal{J},m,t^*} * FutureTransitCost_{i,\mathcal{J},m} \\ *(1 - FutureLongLead_{i,\mathcal{J},m}) \end{pmatrix} \\
&+ \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureProgramStartupCostSchedule_{\mathcal{F},t} \\
&+ \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureProgramActiveCostSchedule_{\mathcal{F},t} \\
&+ \sum_{\substack{\mathcal{J}: \\ t^*=t+FutureLripDelay_{\mathcal{J}}+\eta_{\mathcal{J}}\leq \mathcal{T}}} \begin{pmatrix} bFutureSystem_{\mathcal{J}} * FutureLripCost_{\mathcal{J}} \\ *FutureLripProfile_{\mathcal{F},t^*} * FutureLripLongLead_{\mathcal{J}} \end{pmatrix} \\
&+ \sum_{\substack{\mathcal{J}: \\ t^*=t+FutureLripDelay_{\mathcal{J}}\leq \mathcal{T}}} \begin{pmatrix} bFutureSystem_{\mathcal{J}} * FutureLripCost_{\mathcal{J}} \\ *FutureLripProfile_{\mathcal{F},t^*} * (1 - FutureLripLongLead_{\mathcal{J}}) \end{pmatrix}
\end{aligned} \tag{5.18}$$

- $OSExpense_t$ denotes the total amount spent on O&S of non-future systems at time t within the conventional time horizon.

$$\forall t \leq T$$

$$OSExpense_t = \sum_{(i,m)\in Roles} VehOSCost_{i,m} * NumVehInMission_{i,m,t} \tag{5.19}$$

- $FutureOSExpense_t$ denotes the total amount spent on O&S of future systems at time t in the conventional and extended time horizons and the total amount spent for conventional systems in the extended time horizon.

$\forall t \leq \mathcal{T}$

$$FutureOSExpense_t = \sum_{\substack{i,\mathcal{J},m,t^*: \\ (i,\mathcal{J},m) \in FutureTransitions \\ t^* \leq t}} (NumVehReplaced_{i,\mathcal{J},m,t^*} * VehOSCost_{\mathcal{J}})$$
$$+ \begin{cases} 0 & t \leq T \\ OSExpense_T \\ - \sum\limits_{\substack{i,\mathcal{J},m,t^*: \\ (i,\mathcal{J},m) \in FutureTransitions \\ (i,m) \in Roles \\ T < t^* \leq t}} (NumVehReplaced_{i,\mathcal{J},m,t^*} * VehOSCost_{i,m}) & T < t \leq \mathcal{T} \end{cases} \quad (5.20)$$

- $RdteEffortExpense_{p,t}$ denotes the amount spent on RDT&E by product family p at time t.

$\forall p \in PFWithRdte, t \leq T$

$$RdteEffortExpense_{p,t} = \sum_{d \in AllowedDelays_p} bRdteDelay_{p,d} * RdteCost_{p,d,t} \quad (5.21)$$

- $RdteExpense_t$ denotes the amount spent on all non-future RDT&E efforts at time t within the conventional time horizon.

$\forall t \leq T$

$$RdteExpense_t = \sum_{p \in PFWithRdte} RdteEffortExpense_{p,t} \quad (5.22)$$

- $FutureRdteExpense_t$ denotes the amount spent on RDT&E for all future programs at time t within the conventional and extended time horizons.

$\forall t \leq \mathcal{T}$

$$FutureRdteExpense_t = \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureRdteCostSchedule_{\mathcal{F},t} \quad (5.23)$$

- $CombinedExpense_t$ denotes the combined amount spent on any set of the expenses procurement, O&S, and RDT&E for time period t incurred by non-future systems. Here b_{Proc}, b_{OS}, and b_{Rdte} are user-specified binary indicators that take on value 1 if that expense type is included in the combined expense, and 0 otherwise.

$\forall t \leq T$

$$CombinedExpense_t = b_{Proc} * ProcureExpense_t + b_{OS} * OSExpense_t + b_{Rdte} * RdteExpense_t \quad (5.24)$$

- $FutureCombinedExpense_t$ denotes the combined amount spent on any set of the expenses procurement, O&S, and RDT&E for time period t incurred by future systems.

$$\forall t \leq \mathcal{T}$$

$$FutureCombinedExpense_t = b_{Proc} * FutureProcureExpense_t + b_{OS} * FutureOSExpense_t + b_{Rdte} * FutureRdteExpense_t \tag{5.25}$$

Production

- $NumVehInProduction_{i,t}$ denotes the number of systems of type i that are in a production period at time t. If a particular mission upgrade has no production delay, then delivery is also counted as a production period in those cases. This counts all mission upgrades, storage upgrades, and purchases.

$$\forall i, t \leq T$$

$$\begin{aligned} NumVehInProduction_{i,t} = & \begin{cases} NumVehPurchased_{i,t} & \text{if } PurchProdDelay_i = 0 \\ \sum\limits_{t < t^* \leq t + PurchProdDelay_i} NumVehPurchased_{i,t^*} & \text{if } PurchProdDelay_i > 0 \end{cases} \\ & + \sum_{\substack{j,m,t^*: \\ (j,i,m) \in MissionUpgrades \\ t < t^* \leq t + UpgProdDelay_{j,i} \\ UpgProdDelay_{j,i} > 0}} NumVehInMissionUpgraded_{j,i,m,t^*} \\ & + \sum_{\substack{j,m: \\ (j,i,m) \in MissionUpgrades \\ UpgProdDelay_{j,i} = 0}} NumVehInMissionUpgraded_{j,i,m,t} \\ & + \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpgrades \\ t < t^* \leq t + UpgProdDelay_{j,i} \\ UpgProdDelay_{j,i} > 0}} iNumInStorageUpgraded_{j,i,t^*} \\ & + \sum_{\substack{j: \\ (j,i) \in StorageUpgrades \\ UpgProdDelay_{j,i} = 0}} iNumInStorageUpgraded_{j,i,t} \end{aligned} \tag{5.26}$$

- $NumVehInAdminPeriod_{i,t}$ denotes the number of systems of type i that are in their administrative period at time t. This counts all mission upgrades, storage upgrades, and purchases.

$$\forall i, t \leq T$$

$$\begin{aligned} NumVehInAdminPeriod_{i,t} = & \sum_{t+PurchProdDelay_i < t^* \leq t+PurchDelay_i} NumVehPurchased_{i,t^*} \\ & + \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpgrades \\ t+UpgProdDelay_{j,i} < t^* \leq t+UpgDelay_{j,i}}} iNumInStorageUpgraded_{j,i,t^*} \\ & + \sum_{\substack{j,m,t^*: \\ (j,i,m) \in MissionUpgrades \\ t+UpgProdDelay_{j,i} < t^* \leq t+UpgDelay_{j,i}}} NumVehInMissionUpgraded_{j,i,m,t^*} \end{aligned} \tag{5.27}$$

- $NumVehDelivered_{i,t}$ denotes the number of systems of type i that are delivered (i.e. completed production) in time t via some production facility. This counts all mission upgrades, storage upgrades, and purchases.

$$\forall i, t \leq T$$

$$\begin{aligned} NumVehDelivered_{i,t} = & NumVehPurchased_{i,t} \\ & + \sum_{\substack{j: \\ (j,i) \in StorageUpgrades}} iNumInStorageUpgraded_{j,i,t} \\ & + \sum_{\substack{j,m: \\ (j,i,m) \in MissionUpgrades}} NumVehInMissionUpgraded_{j,i,m,t} \end{aligned} \tag{5.28}$$

Objective Function

The objective function performs different roles depending on the current phase of a mission priority tier. During the "schedule" phase, the objective function **minimizes** the number of schedule violations. Similarly, the "budget" phase objective **minimizes** the number of budget violations given that the number of schedule violation found in the previous phase cannot increase. Running these two phases prior to the performance phase guarantees that, if at all possible, the performance phase is seeded with a business-rule-compliant initial solution. Also, the sequential nature of the schedule and budget phases ensures that violations in schdule and budget cannot be traded off against each other. Since schedule violation minimization occurs first and is then limited to not increase during the budget phase, this assures that budget violations are preferred over schedule violations.

The "performance" phase **maximizes** the cumulative performance of the fleet over the desired planning horizon (either the conventional horizon, or the conventional plus extended horizon if future systems are included) by summing the performance ($\alpha_{i,m}$ parameter) of each system in each mission at each time period. This approach tends to choose modernization schedules that upgrade as many systems as possible as soon as possible so that performance improvements can take effect over as much of the planning horizon as possible. This is a broad characterization however, and the model is also able to avoid early modernization options when it is preferable to wait for even better options in the future. Finally during the "cost" phase, the objective is to **minimize** a user-chosen combination of cumulative procurement, O&S, and RDT&E expenditures while ensuring that the fleet performance attained in the previous phase is not degraded. This will ensure that the smallest cumulative budget is spent in the current tier and maximize the left-over budget that can be used for modernizing lower tiers.

It should also be noted that the formulation source code contains additional variable expressions relating to the objective function that are not documented here. These additional structures are used to easily recreate legacy behavior for troubleshooting and debugging, but are not used by the CPAT tool itself.

- $TierScheduleDeficits$ denotes the sum of all "modernized" or "final mandate" deficit variables throughout the planning horizon. This takes on value zero only if all schedule mandates are met.

$$\begin{aligned} TierScheduleDeficits = & \sum_{\substack{i,m,t: \\ (i,m)\in Roles \\ Tier_m = CurrentTier \\ t\leq T}} iModernizedDeficit_{i,m,t} \\ & + \sum_{\substack{i,m: \\ (i,m)\in Roles \\ Tier_m = CurrentTier}} iFinalMandateDeficit_{i,m} \\ & + \sum_{\mathcal{J}} iFutureSystemMandateDeficit_{\mathcal{J}} \end{aligned} \quad (5.29)$$

- $BudgetOverruns$ denotes the sum of all budget overage amounts for all budget types throughout the conventional and extended planning horizons. This takes on value zero only if all budgets are satisfied.

$$\begin{aligned} BudgetOverruns = & \sum_{t\leq T} (\textit{All Per-Period Budget Overruns}) \\ & + \sum_{t\leq \mathcal{T}} (\textit{All Future Per-Period Budget Overruns}) \\ & + (\textit{All Cumulative Budget Overruns}) \end{aligned} \quad (5.30)$$

- $FuturePerformance$ denotes the cumulative performance of future systems included in the fleet through both conventional and extended time horizons in addition to non-future systems in the extended time horizon.

$$FuturePerformance = \sum_{\substack{i,m,t: \\ (i,m)\in Roles \\ T<t\leq\mathcal{T}}} \alpha_{i,m} * NumVehInMission_{i,m,t} + \sum_{\substack{i,\mathcal{J},m,t: \\ (i,\mathcal{J},m)\in FutureTransitions \\ t\leq\mathcal{T}}} \begin{pmatrix} \alpha_{\mathcal{J},m} * NumVehReplaced_{i,\mathcal{J},m,t} \\ *(\mathcal{T}-t+1) \end{pmatrix} \tag{5.31}$$

- $TotalPerformance$ denotes the cumulative performance of the systems in the fleet. This may include the addition of future systems, if chosen by the user.

$$TotalPerformance = \sum_{\substack{i,m,t: \\ (i,m)\in Roles \\ t\leq T}} \alpha_{i,m} * NumVehInMission_{i,m,t} + FuturePerformance \text{ (if future systems are included)} \tag{5.32}$$

- $CostPhaseExpenses$ denote the combined expenses procurement, O&S, and RDT&E to minimize in the last phase of the optimization for all time periods incurred by non-future systems and future systems. Here $b_{CP-Proc}$, b_{CP-OS}, and $b_{CP-Rdte}$ are user-specified binary indicators that take on value 1 if that expense type is included in the cost phase, and 0 otherwise.

$$\begin{aligned} &CostPhaseExpenses = \\ &\quad \sum_{t\leq T}(b_{CP-Proc} * ProcureExpense_t + b_{CP-OS} * OSExpense_t \\ &\quad + b_{CP-Rdte} * RdteExpense_t) + \sum_{t\leq\mathcal{T}}(b_{CP-Proc} * FutureProcureExpense_t \\ &\quad + b_{CP-OS} * FutureOSExpense_t + b_{CP-Rdte} * FutureRdteExpense_t) \end{aligned} \tag{5.33}$$

- Obj denotes the objective function that either minimizes schedule violations, minimizes budget overruns, maximizes cumulative fleet performance, or minimizes the cumulative combined fleet cost, depending on the current phase. The $SchedulePhase$, $BudgetPhase$, $PerformancePhase$, and $CostPhase$ are binary parameters that indicate which phase is currently being optimized. One of these parameters is always 1, while the rest are 0.

$$\begin{aligned} Obj = &-SchedulePhase * TierScheduleDeficits \\ &- BudgetPhase * 0.00001 * BudgetOverruns \\ &+ PerformancePhase * TotalPerformance \\ &- CostPhase * 0.00001 * CostPhaseExpenses \end{aligned} \tag{5.34}$$

Chapter 6

MILP Constraints

Multi-Tier, Multi-Phase Constraints

- For phases after the schedule phase, limit the amount of schedule violation so that it cannot increase from the violation amount reported in the schedule phase. During the schedule phase, the $TierScheduleDeficitBound$ parameter is not restrictive. This partially addresses the **Tier Phases** business rule.

$$TierScheduleDeficits \leq TierScheduleDeficitBound + 0.001 \tag{6.1}$$

- For phases after the budget phase, limit the amount of budget overrun so that it cannot increase from the overages reported in the budget phase. Prior to and during the budget phase, the $BudgetOverrunBound$ parameter is not restrictive. This partially addresses the **Tier Phases** business rule.

$$BudgetOverruns \leq BudgetOverrunBound + 0.001 \tag{6.2}$$

- For phases after the performance phase, ensure that the cumulative fleet performance does not degrade from the value found in the performance phase. Prior to and during the performance phase, the $MinimumPerformance$ parameter is not restrictive. This partially addresses the **Tier Phases** business rule.

$$TotalPerformance >= MinimumPerformance - 0.001 \tag{6.3}$$

- Constraints (6.4)–(6.7) ensure that no modernization occurs for missions in lower-priority tiers than the current tier and address the **Priority Tiers** business rule.

$$\forall (i,j,m) \in MissionUpgrades, t \leq T \text{ where } Tier_m > CurrentTier$$
$$NumVehInMissionUpgraded_{i,j,m,t} = 0 \tag{6.4}$$

$$\forall (i,j,m) \in Transitions, t \leq T \text{ where } Tier_m > CurrentTier$$
$$NumVehFromStorage_{i,j,m,t} = 0 \tag{6.5}$$

$$\forall (i,m) \in Roles, t \leq \mathcal{T} \text{ where } Tier_m > CurrentTier$$
$$iModernizedDeficit_{i,m,t} = 0 \tag{6.6}$$

$$\forall (i,m) \in Roles \text{ where } Tier_m > CurrentTier$$
$$iFinalMandateDeficit_{i,m} = 0 \quad (6.7)$$

- Constraints (6.8)–(6.11) ensure that the modernization schedules previously determined for higher-priority tiers continue to be held while optimizing lower priority tiers, also addressing the **Priority Tiers** business rule.

$$\forall (i,j,m,t,N) \in fixedVehInMissionUpgraded$$
$$iNumBrigInMissionUpgraded_{i,j,m,t} = N \quad (6.8)$$

$$\forall (i,j,m,t,N) \in fixedVehFromStorage$$
$$iNumBrigFromStorage_{i,j,m,t} = N \quad (6.9)$$

$$\forall (i,m,t,N) \in fixedModernizedDeficit$$
$$iModernizedDeficit_{i,m,t} = N \quad (6.10)$$

$$\forall (i,m,N) \in fixedFinalMandateDeficit$$
$$iFinalMandateDeficit_{i,m} = N \quad (6.11)$$

- Constraints (6.12)–(6.14) ensure that if one mission precedes another, than the succeeding mission cannot field until the preceding mission has 1) finished fielding and 2) modernized 100% of its initial systems. This fulfills the **Mission Succession** business rule.

$$\forall m \in PrecededMissions, t \leq T$$
$$\sum_{i,j:\ (i,j,m) \in Transitions} iNumBrigFromStorage_{i,j,m,t} + \sum_{i,j:\ (i,j,m) \in MissionUpgrades} iNumBrigInMissionUpgraded_{i,j,m,t} \leq maxYearlyBrigTransitLimit * bMissionCanField_{m,t} \quad (6.12)$$

$\forall m \in PrecededMissions, t \leq T$

$$
\begin{aligned}
& 1 - bMissionCanField_{m,t} \\
\leq & \sum_{\substack{i,m^*: \\ (i,m^*)\in Roles \\ (m^*,m)\in MissionSuccessions \\ InitialBrigInMission_{i,m^*}>0}} NumBrigInMission_{i,m^*,t} \\
+ & \sum_{\substack{i,j,m^*,t^*: \\ (i,j,m^*)\in Transitions \\ (m^*,m)\in MissionSuccessions \\ t\leq t^*\leq T}} iNumBrigFromStorage_{i,j,m^*,t^*} \\
+ & \sum_{\substack{i,j,m^*,t^*: \\ (i,j,m^*)\in MissionUpgrades \\ (m^*,m)\in MissionSuccessions \\ t\leq t^*\leq T}} iNumBrigInMissionUpgraded_{i,j,m^*,t^*}
\end{aligned}
\quad (6.13)
$$

$\forall m \in PrecededMissions, t \leq T$

$$
\begin{aligned}
& \left(\sum_{m^*:\ (m^*,m)\in MissionSuccessions} (BrigPerMission_{m^*} * (maxPathLength_{m^*} + 1)) \right) \\
& * (1 - bMissionCanField_{m,t}) \\
\geq & \sum_{\substack{i,m^*: \\ (i,m^*)\in Roles \\ (m^*,m)\in MissionSuccessions \\ InitialBrigInMission_{i,m^*}>0}} NumBrigInMission_{i,m^*,t} \\
+ & \sum_{\substack{i,j,m^*,t^*: \\ (i,j,m^*)\in Transitions \\ (m^*,m)\in MissionSuccessions \\ t\leq t^*\leq T}} iNumBrigFromStorage_{i,j,m^*,t^*} \\
+ & \sum_{\substack{i,j,m^*,t^*: \\ (i,j,m^*)\in MissionUpgrades \\ (m^*,m)\in MissionSuccessions \\ t\leq t^*\leq T}} iNumBrigInMissionUpgrade_{i,j,m^*,t^*}
\end{aligned}
\quad (6.14)
$$

System Flow Conservation Constraints

- This constraint implies that the number of systems i in storage at time t, less the ones already spoken for, must always be at least as many as how many are taken out at t. This fulfills the storage part of the **Outflow Availability** business rule.

$\forall i, t \leq T$

$$NumVehInStorageExchangeable_{i,t} \geq 0 \quad (6.15)$$

- This constraint ensures that the number of systems i in mission m at time t (not counting the ones that have been spoken for by future mission upgrades) is nonnegative. This helps fulfill the mission part of the **Outflow Availability** business rule.

$$\forall (i,m) \in Roles, t \leq T$$

$$NumVehMissionExchangeable_{i,m,t} \geq 0 \quad (6.16)$$

- This constraint limits the pool of potential systems i that can be upgraded in storage at time t (purchased i's are not included in this pool). This partially prevents newly purchased systems in storage from being upgraded before they are sent to mission, thus addressing the **No Pre-Usage Upgrades** business rule.

$$\forall i, t \leq T$$

$$\sum_{\substack{j,t^*: \\ (i,j) \in StorageUpgrades \\ t^* \leq t + UpgProdDelay_{i,j}}} iNumInStorageUpgraded_{i,j,t^*} \leq$$

$$InitialVehInStorage_i$$

$$+ \sum_{\substack{j,m,t^*: \\ (i,j,m) \in Transitions \\ t^* \leq t}} NumVehFromStorage_{i,j,m,t^*}$$

$$+ \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpgrades \\ t^* \leq t}} iNumInStorageUpgraded_{j,i,t^*}$$

$$+ \sum_{\substack{\mathcal{J},m,t^*: \\ (i,\mathcal{J},m) \in FutureTransitionPurchases \\ t^* \leq t}} NumVehReplaced_{i,\mathcal{J},m,t^*} \quad (6.17)$$

- This constraint is only used if pre-purchasing is turned off and ensures that all systems i purchased up in-storage-upgraded to at time period t must be fielded to some mission m in that same time period. This addresses the **Optional Pre-Purchasing** business rule.
  If $AllowPrePurchasing = 0$

$$\forall i, t \leq T$$

$$NumVehPurchased_{i,t}$$

$$+ \sum_{\substack{j: \\ (j,i) \in StorageUpgrades}} iNumInStorageUpgraded_{j,i,t} \leq$$

$$\sum_{\substack{j,m: \\ (j,i,m) \in Transitions}} NumVehFromStorage_{j,i,m,t} \quad (6.18)$$

- Constraints (6.19) and (6.20) ensure that any group of systems i that are retired from some mission m in time period t cannot be immediately re-fielded back to the same mission in the same time period. This fulfills the **No Retire and Re-Field** business rule.

$$\forall (i,m) \in InterimRoles, t \leq T$$
$$YearlyBridgadeTransitLimit_m * bInterimBrigCanField_{i,m,t} \geq \sum_{j \in Inflow_{i,m}} iNumBrigFromStorage_{j,i,m,t} \quad (6.19)$$

$$\forall (i,m) \in InterimRoles, t \leq T$$
$$YearlyBridgadeTransitLimit_m * (1 - bInterimBrigCanField_{i,m,t}) \geq \sum_{j \in Outflow_{i,m}} iNumBrigFromStorage_{i,j,m,t} \quad (6.20)$$

General Scheduling Constraints

- Constraints (6.21)–(6.23) ensure that at the beginning of the planning horizon, no systems are purchased, in-mission upgraded, or in-storage upgraded earlier than the length of the associated delivery delay (plus an extra year if there is an accompanying long lead). If this was not done, then costs could be incurred prior to the beginning of the planning horizon. These fulfill the **Early Transition Charging** business rule.

$$\forall i \in PurchasableVeh, t \leq \begin{cases} PurchDelay_i + 1 & \text{if } PrePurchCost_i > 0 \\ PurchDelay_i & \text{if } PrePurchCost_i = 0 \end{cases}$$
$$iNumBatchesPurchased_{i,t} = 0 \quad (6.21)$$

$$\forall (i,j,m) \in MissionUpgrades, t \leq \begin{cases} UpgDelay_{i,j} + 1 & \text{if } PreUpgCost_{i,j} > 0 \\ UpgDelay_{i,j} & \text{if } PreUpgCost_{i,j} = 0 \end{cases}$$
$$iNumBrigInMissionUpgraded_{i,j,m,t} = 0 \quad (6.22)$$

$$\forall (i,j) \in StorageUpgrades, t \leq \begin{cases} UpgDelay_{i,j} + 1 & \text{if } PreUpgCost_{i,j} > 0 \\ UpgDelay_{i,j} & \text{if } PreUpgCost_{i,j} = 0 \end{cases}$$
$$iNumInStorageUpgraded_{i,j,t} = 0 \quad (6.23)$$

- This constraint ensures that the required percentage of initial systems i in mission m are retired (i.e. transitioned out) by time t. If it is not possible to retire the required percentage due to other constraints, then this deficit is captured by the

$iModernizedDeficit_{i,m,t}$ variables. This fulfills the **System Modernization Requirements** business rule.

$$\forall (i,m) \in Roles, t \leq \mathcal{T} \text{ where } ModernPercent_{i,m,t} > 0 \text{ and } Tier_m = CurrentTier$$
$$CumulativeBrigRetiredFromRole_{i,m,t} * VehPerBrigade_m + iModernizedDeficit_{i,m,t} \geq ModernPercent_{i,m,t} * InitialVehInMission_{i,m} \quad (6.24)$$

- This constraint ensures that the number of groups transitioned for mission m at time t is below a specified limit, fulfilling the **Per-Period Mission Modernization Limit** business rule.

$$\forall m, t \leq T \text{ where } YearlyBrigTransitLimit_m < \infty \text{ and } Tier_m = CurrentTier$$
$$\sum_{i,j:\ (i,j,m) \in Transitions} NumBrigTransit_{i,j,m,t} \leq YearlyBrigTransitLimit_m \quad (6.25)$$

- This constraint ensures that the cumulative number of groups of initial systems modernized for mission m throughout the conventional and extended planning horizon is below a specified limit. This fulfills the **Cumulative Mission Modernization Limit** business rule.

$$\forall m \text{ where } CumulativeBrigTransitLimit_m < \infty \text{ and } Tier_m = CurrentTier$$
$$\sum_{\substack{i,j,t: \\ (i,j,m) \in Transitions \\ InitialVehInMission_{i,m} > 0 \\ t \leq T}} NumBrigTransit_{i,j,m,t}$$
$$+ \sum_{\substack{i,\mathcal{J},t: \\ (i,\mathcal{J},m) \in FutureTransitions \\ InitialVehInMission_{i,m} > 0 \\ t \leq \mathcal{T}}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq CumulativeBrigTransitLimit_m \quad (6.26)$$

- This constraint ensures that the number of systems i in mission m meets or exceeds a certain level by time $\mathcal{T}$, fulfilling the **System Mandates** business rule.

$$\forall (i,m) \in MandatedRoles \text{ where } Tier_m = CurrentTier$$
$$NumVehInMission_{i,m,\mathcal{T}} + iFinalMandateDeficit_{i,m} * VehPerBrigade_m \geq FinalMandate_{i,m} \quad (6.27)$$

- This constraint ensures for system obviation pairs (i,j) that system j can be delivered only if system i has not been delivered at time t or earlier. In other words, j deliveries can only occur before i deliveries. This fulfills the **System Obviation** business rule.

$$\forall (i,j) \in SystemObviations, t, t^* \text{ where } i,j \in DeliverableVeh \text{ and } t^* \leq t \leq T$$
$$bVehDelivered_{j,t} <= 1 - bVehDelivered_{i,t^*} \quad (6.28)$$

- This constraint ensures that for each synchronization set s, the number of groups of synchronized systems in the synchronized mission is always equal. This fulfills the **Synchronization Sets** business rule. Note that for any set, such as $Z = \{z_1, z_2, z_3\}$, we have the syntax that $First(Z) = z_1$.

$$\forall s, m \in SyncMissions_s, t \leq T \text{ where } m \neq First(SyncMissions_s)$$
$$\sum_{\substack{i,m^*: \\ (i,m^*)\in Roles \\ m^*=First(SyncMissions_s) \\ i\in SyncSystems_s}} NumBrigInMission_{i,m^*,t} = \sum_{\substack{i: \\ (i,m)\in Roles \\ i\in SyncSystems_s}} NumBrigInMission_{i,m,t} \tag{6.29}$$

- Constraints (6.30)–(6.32) properly set the value of the $bVehInStorageExchangeable_{i,t}$ indicator variable and then enforce the **Storage Consumption Priority** business rule.

$$\forall i, t \leq T \text{ where } \exists(i,j) \in StoragePriorityPairs$$
$$bVehInStorageExchangeable_{i,t} \leq NumVehInStorageExchangeable_{i,t} \tag{6.30}$$

$$\forall i, t \leq T \text{ where } \exists(i,j) \in StoragePriorityPairs$$
$$TotalVehPopulation * bVehInStorageExchangeable_{i,t} \geq NumVehInStorageExchangeable_{i,t} \tag{6.31}$$

$$\forall i, t \leq T \text{ where } \exists(i,j) \in StoragePriorityPairs$$
$$\sum_{\substack{j,j^*: \\ (i,j)\in StoragePriorityPair \\ (j,j^*)\in StorageUpgrades \\ t+UpgProdDelay_{j,j^*}\leq T}} iNumInStorageUpgraded_{j,j^*,t+UpgProdDelay_{j,j^*}} \leq TotalVehPopulation^2 * (1 - bVehInStorageExchangeable_{i,t}) \tag{6.32}$$

- Constraints (6.33) – (6.35) properly set $bUpgBeforePurchInStorageExchangeable_{i,t}$ and then enforce the **Upgrades Trump Purchases** business rule.

$$\forall i \in UpgBeforePurch, t \leq T$$
$$bUpgBeforePurchInStorageExchangeable_{i,t} \leq NumVehInStorageExchangeable_{i,t} \tag{6.33}$$

$$\forall i \in UpgBeforePurch, t \leq T$$
$$TotalVehPopulation * bUpgBeforePurchInStorageExchangeable_{i,t} \geq NumVehInStorageExchangeable_{i,t} \tag{6.34}$$

$\forall i \in UpgBeforePurch, t \leq T$

$$\sum_{\substack{j: \\ (i,j) \in StorageUpgrades \\ j \in PurchasableVeh \\ t+PurchProdDelay_j \leq T}} iNumBatchesPurchased_{j,t+PurchProdDelay_j} \leq$$

$$TotalVehPopulation * (1 - bUpgBeforePurchInStorageExchangeable_{i,t}) \quad (6.35)$$

Budget Constraints

- Constraints (6.36) – (6.39) combine to fulfill the **Per-Period Budgets** business rule for time periods in the conventional time horizon. Note that some future programs may incur costs during the conventional time horizon that must be accounted for. Also note that if other constraints throughout the formulation force a particular per-period budget to be violated, then the amount of overage is determined by the appropriate budget constraint and stored in the "Overrun" variables. These variables can help the analyst pin-point where particular business rule violations arise due to overly restrictive input parameters.

$\forall t \leq T$ where $ProcureBudget_t < \infty$

$$ProcureExpense_t + FutureProcureExpense_t \leq$$
$$ProcureBudget_t + fProcureBudgetOverrun_t \quad (6.36)$$

$\forall t \leq T$ where $OSBudget_t < \infty$

$$OSExpense_t + FutureOSExpense_t \leq$$
$$OSBudget_t + fOSBudgetOverrun_t \quad (6.37)$$

$\forall t \leq T$ where $RdteBudget_t < \infty$

$$RdteExpense_t + FutureRdteExpense_t \leq$$
$$RdteBudget_t + fRdteBudgetOverrun_t \quad (6.38)$$

$\forall t \leq T$ where $CombinedBudget_t < \infty$

$$CombinedExpense_t + FutureCombinedExpense_t \leq$$
$$CombinedBudget_t + fCombinedBudgetOverrun_t \quad (6.39)$$

- Constraints (6.40) – (6.43) combine to fulfill the **Per-Period Budgets** business rule in the extended time horizon. They operate in a manner similar to the previous constraints above, but only need to limit expenses incurred in the extended time horizon.

$\forall\ T < t \leq \mathcal{T}$ where $FutureProcureBudget_t < \infty$

$$FutureProcureExpense_t \leq$$
$$FutureProcureBudget_t + fFutureProcureBudgetOverrun_t \quad (6.40)$$

$\forall\ T < t \leq \mathcal{T}$ where $FutureOSBudget_t < \infty$

$$FutureOSExpense_t \leq FutureOSBudget_t + fFutureOSBudgetOverrun_t \quad (6.41)$$

$\forall\ T < t \leq \mathcal{T}$ where $FutureRdteBudget_t < \infty$

$$FutureRdteExpense_t \leq FutureRdteBudget_t + fFutureRdteBudgetOverrun_t \quad (6.42)$$

$\forall\ T < t \leq \mathcal{T}$ where $FutureCombinedBudget_t < \infty$

$$FutureCombinedExpense_t \leq FutureCombinedBudget_t + fFutureCombinedBudgetOverrun_t \quad (6.43)$$

- Constraints (6.44) – (6.47) combine to fulfill the **Cumulative Budgets** business rule. "Overrun" variables are used here in a similar manner to the previous per-period constraints. Note that a cumulative budget applies both to the future <u>and</u> non-future system expenses across both the conventional <u>and</u> extended time horizons.

if $CumulativeProcureBudget < \infty$

$$\sum_{t \leq T} ProcureExpense_t + \sum_{t \leq \mathcal{T}} FutureProcureExpense_t \leq CumulativeProcureBudget + fCumulativeProcureBudgetOverrun \quad (6.44)$$

if $CumulativeOSBudget < \infty$

$$\sum_{t \leq T} OSExpense_t + \sum_{t \leq \mathcal{T}} FutureOSExpense_t \leq CumulativeOSBudget + fCumulativeOSBudgetOverrun \quad (6.45)$$

if $CumulativeRdteBudget < \infty$

$$\sum_{t \leq T} RdteExpense_t + \sum_{t \leq \mathcal{T}} FutureRdteExpense_t \leq CumulativeRdteBudget + fCumulativeRdteBudgetOverrun \quad (6.46)$$

if $CumulativeCombinedBudget < \infty$

$$\sum_{t \leq T} CombinedExpense_t + \sum_{t \leq \mathcal{T}} FutureCombinedExpense_t \leq CumulativeCombinedBudget + fCumulativeCombinedBudgetOverrun \quad (6.47)$$

Group Density Levels

- Constraints (6.48) and (6.49) ensure that the $bTransitionedToDensityLevel_{i,m,\ell}$ indicator variable equals 1 if and only if there are ever any transitions to system i in mission m and those transitions achieve a density level of $\ell \in UpgDensityLevels_m$ groups. These constraints partially fulfill the **Minimum Group Transition Density** business rule.

$$\forall (i,m) \text{ where } \exists (i,m,\ell) \in UpgDensityFlags$$

$$\sum_{\substack{j,t:\\(j,i,m)\in Transitions\\ t\le T}} NumBrigTransit_{j,i,m,t} \geq \sum_{\ell \in UpgDensityLevels_m} (bTransitionedToDensityLevel_{i,m,\ell} * \ell) \quad (6.48)$$

$$\forall (i,m) \text{ where } \exists (i,m,\ell) \in UpgDensityFlags$$

$$\sum_{\substack{j,t:\\(j,i,m)\in Transitions\\ t\le T}} NumBrigTransit_{j,i,m,t} \leq \sum_{\substack{\ell:\\ \ell\in UpgDensityLevels_m\\ \ell\neq \max(UpgDensityLevels_m)}} (bTransitionedToDensityLevel_{i,m,\ell} * \ell) + \sum_{\substack{\ell:\\ \ell\in UpgDensityLevels_m\\ \ell= \max(UpgDensityLevels_m)}} (bTransitionedToDensityLevel_{i,m,\ell} * BrigPerMission_m) \quad (6.49)$$

- Constraint (6.50) ensures that system i in mission m can satisfy at most 1 of the minimum transition density levels. Note that if system i never transitions into system m at any time, then all three of the $bTransitionedToDensityLevel$ binaries will be 0. Together with (6.48) and (6.49), this fulfills the **Minimum Group Transition Density** business rule.

$$\forall (i,m) \text{ where } \exists (i,m,\ell) \in UpgDensityFlags$$

$$\sum_{\ell \in UpgDensityLevels_m} bTransitionedToDensityLevel_{i,m,\ell} \leq 1 \quad (6.50)$$

- Constraints (6.51) and (6.52) ensure the $bHasFinalDensity_{i,m,\ell}$ indicator variable equals 1 if and only if system i in mission m has density $\ell \in FinalDensityLevels_m$ groups at time $\mathcal{T}$. These constraints partially fulfill the **Minimum Group Final**

Density business rule.

$$\forall (i,m,\ell) \in FinalDensityFlags$$
$$NumBrigInMission_{i,m,\mathcal{T}} \geq (bHasFinalDensity_{i,m,\ell} + TransitionedToRole_{i,m} - 1) * \ell - \begin{cases} CumulativeBrigRetiredFromRole_{i,m,\mathcal{T}} * \ell & \text{if } \ell = \max(FinalDensityLevels_m) \\ 0 & \text{otherwise} \end{cases} \tag{6.51}$$

$$\forall (i,m) \text{ where } \exists (i,m,\ell) \in FinalDensityFlags$$
$$NumBrigInMission_{i,m,\mathcal{T}} \leq \sum_{\substack{\ell: \\ \ell \in FinalDensityLevels_m \\ \ell \neq \max(FinalDensityLevels_m)}} (bHasFinalDensity_{i,m,\ell} * \ell) + \sum_{\substack{\ell: \\ \ell \in FinalDensityLevels_m \\ \ell = \max(FinalDensityLevels_m)}} (bHasFinalDensity_{i,m,\ell} * BrigPerMission_m) \tag{6.52}$$

- Constraint (6.53) ensures that system i in mission m can satisfy at most 1 of the final transition density levels. Note that if system i never transitions into system m at any time, then all three of the $bHasFinalDensity$ binaries will be 0. Together with (6.51) and (6.52), this fulfills the **Minimum Group Final Density** business rule.

$$\forall (i,m) \text{ where } \exists (i,m,\ell) \in FinalDensityFlags$$
$$\sum_{\ell \in FinalDensityLevels_m} bHasFinalDensity_{i,m,\ell} \leq 1 \tag{6.53}$$

System Production Constraints

- Constraints (6.54) and (6.55) ensure that the $bVehDelivered$ flag for each time period is 1 if and only if systems are delivered in that period. This flag is then used to help fulfill a variaty of business rules.

$$\forall i \in DeliverableVeh, t \leq T$$
$$bVehDelivered_{i,t} \leq NumVehDelivered_{i,t} \tag{6.54}$$

$$\forall i \in DeliverableVeh, t \leq T$$
$$TotalVehPopulation * bVehDelivered_{i,t} \geq NumVehDelivered_{i,t} \tag{6.55}$$

- If a system i is not a free interim upgrade, then this constraint ensures that i cannot be delivered at time t unless it is also fielded to a mission at some other time. This avoids unnecessary production costs and fulfills the **Delivery Implies Fielding** business rule.

$$\forall i, t \leq T \text{ where } i \in DeliverableVeh \text{ and } i \notin FreeInterimUpgVeh$$
$$\sum_{\substack{j,m,t^*: \\ (j,i,m) \in Transitions \\ t^* \leq T}} NumBrigTransit_{j,i,m,t^*} \geq bVehDelivered_{i,t} \quad (6.56)$$

- Constraints (6.57) and (6.58) ensure that the $bVehEverDelivered_i$ flag is 1 if and only if system i ever delivered throughout the conventional planning horizon. This flag will then determine which LRIP profiles to activate.

$$\forall i \in DeliverableVeh, t \leq T$$
$$bVehEverDelivered_i \geq bVehDelivered_{i,t} \quad (6.57)$$

$$\forall i \in DeliverableVeh$$
$$bVehEverDelivered_i \leq \sum_{t \leq T} bVehDelivered_{i,t} \quad (6.58)$$

Product Family Constraints

- Constraints (6.59) and (6.60) ensure that for each product family having a minimum production rate, the $bProductFamilyDelivered_{p,t}$ indicator variable is 1 if and only if at least one of the member systems of p is delivered at time t. This flag helps support the **Minimum Sustaining Rate** and **Production Smoothing** business rules.

$$\forall p \in PFWithProdCtrls, t \leq T$$
$$bProductFamilyDelivered_{p,t} \leq \sum_{i \in ProductFamily_p \cap DeliverableVeh} bVehDelivered_{i,t} \quad (6.59)$$

$$\forall p \in PFWithProdCtrls, i \in DeliverableVeh, t \leq T$$
$$bProductFamilyDelivered_{p,t} \geq bVehDelivered_{i,t} \quad (6.60)$$

- Constraints (6.61) and (6.62) ensure that for each product family p having an active cost and for each t, the $bProductFamilyActive_{p,t}$ flag is 1 if and only if some member systems of p are in production or administrative periods (for either LRIP or FRP) at time t. This flag supports the **Active Product Families** and **Family Per-Period**

Costs business rules.

$$\forall p \in PFWithActive, t \leq T$$
$$bProductFamilyActive_{p,t} \leq \sum_{i \in ProductFamily_p} NumVehInProduction_{i,t} + \sum_{i \in ProductFamily_p} NumVehInAdminPeriod_{i,t} + \sum_{i \in ProductFamily_p} NumLripVehActive_{i,t} \quad (6.61)$$

$$\forall p \in PFWithActive, t \leq T$$
$$10 * TotalVehPopulation * bProductFamilyActive_{p.t} \geq \sum_{i \in ProductFamily_p} NumVehInProduction_{p,t} + \sum_{i \in ProductFamily_p} NumVehInAdminPeriod_{i,t} + \sum_{i \in ProductFamily_p} NumLripVehActive_{i,t} \quad (6.62)$$

- Constraints (6.63)–(6.65) ensure that for each product family p having a start-up cost profile, the flag $bProductFamilyStartup_{p,t}$ is 1 if and only if time t is the first time that a member system of p enters a FRP delay period. This helps fulfill the **Family Start-Up Costs** business rule.

$$\forall p \in PFWithStartup, t \leq T$$
$$bProductFamilyStartup_{p,t} \leq \sum_{i \in ProductFamily_p} (NumVehInProduction_{i,t} + NumVehInAdminPeriod_{i,t}) \quad (6.63)$$

$$\forall p \in PFWithStartup, t \leq T$$
$$TotalVehPopulation * \sum_{t^* \leq t} bProductFamilyStartup_{p,t^*} \geq \sum_{i \in ProductFamily_p} (NumVehInProduction_{i,t} + NumVehInAdminPeriod_{i,t}) \quad (6.64)$$

$$\forall p \in PFWithStartup$$
$$\sum_{t \leq T} bProductFamilyStartup_{p,t} \leq 1 \quad (6.65)$$

- Constraints (6.66)–(6.68) ensure that for each product family p having LRIP, the $bProductFamilyFrpStarted_{p,t}$ flag is 1 if and only if time t is the first time that a member system of p delivers FRP assets. This helps fulfill the **LRIP Timing** business rule.

$$\forall p \in PFWithLrip, t \leq T$$
$$bProductFamilyFrpStarted_{p,t} \leq \sum_{i \in ProductFamily_p \cap DeliverableVeh} bVehDelivered_{i,t} \tag{6.66}$$

$$\forall p, i, t \leq T \text{ where } p \in PFWithLrip \text{ and } i \in ProductFamily_p \cap DeliverableVeh$$
$$\sum_{t^* \leq t} bProductFamilyFrpStarted_{p,t^*} \geq bVehDelivered_{i,t} \tag{6.67}$$

$$\forall p \in PFWithLrip$$
$$\sum_{t \leq T} bProductFamilyFrpStarted_{p,t} \leq 1 \tag{6.68}$$

- This constraint ensures that for each product family p that disallows gaps, all systems in the family must be delivered during a set of contiguous time periods. This satisfies the **Delivery Gaps** business rule.

$$\forall p, t, t^* \text{ where } ProdFamilyAllowGaps_p = 0 \text{ and } t^* + 1 < t \leq T$$
$$\begin{aligned} bProductFamilyDelivered_{p,t} - bProductFamilyDelivered_{p,t-1} \\ + bProductFamilyDelivered_{p,t^*} \leq 1 \end{aligned} \tag{6.69}$$

- Constraints (6.70) and (6.71) ensure that each product family p having a startup profile can only become active at times when the entire start-up cost profile would be incurred (i.e. no parts of the cost profile occur before or after the planning horizon). This partially fulfills the **Early/Late Transition Charging** business rule.

$$\forall p \in PFWithStartup, t \leq T \text{ where } \sum_{t \leq t^* < T} ProdFamilyStartupCostSched_{p,-t^*} > 0$$
$$bProductFamilyStartup_{p,t} = 0 \tag{6.70}$$

$$\forall p \in PFWithStartup, t \leq T \text{ where } \sum_{T-t < t^* < T} ProdFamilyStartupCostSched_{p,t^*} > 0$$
$$bProductFamilyStartup_{p,t} = 0 \tag{6.71}$$

- This constraint ensures that the number of systems delivered each time period by each product family is less than the specified capacity. This fulfills this **Family Per-Period Capacity** business rule.

$$\forall p, t \leq T \text{ where } ProdFamilyMaxDelivery_{p,t} < \infty$$
$$\sum_{i \in ProductFamily_p} NumVehDelivered_{i,t} \leq ProdFamilyMaxDelivery_{p,t} \quad (6.72)$$

- This constraint ensures that the cumulative number of systems delivered by each product family is less than the specified capacity. This fulfills this **Family Cumulative Capacity** business rule.

$$\forall p \text{ where } ProdFamilyMaxCumulativeDelivery_p < \infty$$
$$\sum_{\substack{i,t: \\ i \in ProductFamily_p \\ t \leq T}} NumVehDelivered_{i,t} \leq ProdFamilyMaxCumulativeDelivery_p \quad (6.73)$$

- This constraint ensures that if systems are delivered from product family p at time t, then the number of systems delivered must at least meet the minimum sustaining rate for that family. This fulfills the **Minimum Sustaining Rate** business rule.

$$\forall p, t < T \text{ where } 1 < ProdFamilyMinDelivery_p < ProdFamilyMaxDelivery_{p,t}$$
$$\sum_{i \in ProductFamily_p} NumVehDelivered_{i,t} \geq$$
$$bProductFamilyDelivered_{p,t} * ProdFamilyMinDelivery_p$$
$$- (1 - bProductFamilyDelivered_{p,t+1}) * TotalVehPopulation \quad (6.74)$$

- Constraints (6.75)–(6.78) combine to enforce the **RDT&E Costs** business rule when non-zero RDT&E delays are allowed. They are enforced <u>if and only if</u> the delays <u>are</u> allowed.

  If $EnableRdteDelays = 1$

$$\forall p \in PFWithRdte, d$$
$$bRdteDelay_{p,d} \leq \sum_{\substack{i: \\ i \in ProductFamily_p \cap DeliverableVeh \\ FirstAvailable_i + d \leq T}} bVehDelivered_{i,FirstAvailable_i + d} \quad (6.75)$$

$$\forall p \in PFWithRdte, d, i \text{ where } i \in DeliverableVeh \text{ and } FirstAvailable_i + d \leq T$$
$$\sum_{d^* \leq d} bRdteDelay_{p,d^*} \geq bVehDelivered_{i,FirstAvailable_i + d} \quad (6.76)$$

$\forall p \in PFWithRdte$

$$\sum_d bRdteDelay_{p,d} \leq 1 \quad (6.77)$$

$\forall p \in PFWithRdte, d \notin AllowedDelays_p$

$$\sum_d bRdteDelay_{p,d} = 0 \quad (6.78)$$

- Constraints (6.79)–(6.81) combine to enforce legacy RDT&E behavior if the parameter $EnableRdteDelays = 0$. Note that this does not imply that systems in a product family with RDT&E cost profiles can only start on time or not at all. Instead, legacy behavior allows systems in the product family to start at any time, as long as the 0-delay cost profile is incurred.

If $EnableRdteDelays = 0$

$\forall p \in PFWithRdte, d > 0$

$$bRdteDelay_{p,d} = 0 \quad (6.79)$$

$\forall p \in PFWithRdte$

$$bRdteDelay_{p,0} \leq \sum_{\substack{i,t:\\ i \in ProductFamily_p \cap DeliverableVeh\\ t \leq T}} bVehDelivered_{i,t} \quad (6.80)$$

$\forall p \in PFWithRdte, i, t \leq T$ where $i \in ProductFamily_p \cap DeliverableVeh$

$$bRdteDelay_{p,0} \geq bVehDelivered_{i,t} \quad (6.81)$$

LRIP Constraints

- Constraints (6.82)–(6.84) ensure that the $bLripVehBaseYear_{p,i,t}$ flag is 1 if and only if 1) product family p enters full-rate production at time t and, 2) system i is ever delivered. This ensures that LRIP profiles from different systems in a product family will all line up with the beginning of FRP for that family, fulfilling the **LRIP Timing** business rule.

$\forall (p, i) \in LripProfiles, t \leq T$

$$bLripVehBaseYear_{p,i,t} \leq bProductFamilyFrpStarted_{p,t} \quad (6.82)$$

$\forall (p, i) \in LripProfiles, t \leq T$

$$bLripVehBaseYear_{p,i,t} \leq bVehEverDelivered_i \quad (6.83)$$

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$bLripVehBaseYear_{p,i,t} \geq bVehEverDelivered_i + bProductFamilyFrpStarted_{p,t} - 1 \tag{6.84}$$

- This constraint ensures that for all systems i in product family p with an LRIP profile cannot begin full-rate production for those systems until all delays are complete. Here, the binary parameter $\psi_{p,i}$ takes value 1 if $LripPreCost_{p,i} > 0$ and 0 otherwise. This partially satisfies the **Early/Late Transition Charging** business rule.

$$\forall (p,i) \in LripProfiles, t^* \in LripYears, t \text{ where } LripPreProduction_{p,i,t^*} > 0 \text{ and}$$
$$t <= t^* + LripDelay_{p,i} + \psi_{p,i}$$
$$bLripVehBaseYear_{p,i,t} = 0 \tag{6.85}$$

Production Smoothing Constraints

- These constraints ensure that the total number of systems i produced for product family p, when that product family is in production, for each time period t is within a certain variance of that family's median production level. This fulfills the **Production Smoothing** business rule.

$$\forall p, t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p < t \leq T$$
$$\sum_{i \in ProductFamily_p} NumVehDelivered_{i,t} \leq$$
$$(1 + 0.5 * MaxDeliveryVariance_p) * fMedianDeliveryLevel_p \tag{6.86}$$

$$\forall p, t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p < t < T - 1$$
$$\sum_{i \in ProductFamily_p} NumVehDelivered_{i,t} \geq$$
$$(1 - 0.5 * MaxDeliveryVariance_p) * fMedianDeliveryLevel_p$$
$$- \sum_{t^* \in \{0,\ldots,RampUp_p\}} (1 - bProductFamilyDelivered_{p,t-t^*}) * TotalVehPopulation$$
$$- (1 - bProductFamilyDelivered_{p,t+1}) * TotalVehPopulation \tag{6.87}$$

- This constraint ensures that for each product family p having a ramp-up, the number of systems delivered during ramp-up is non-decreasing. This fulfills the **Production**

Ramp-up business rule.

$$\forall p,t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p > 0 \text{ and } 1 < t \leq T$$

$$\sum_{i \in ProductFamily_p} NumVehDelivered_{i,t} \geq \sum_{i \in ProductFamily_p} NumVehDelivered_{i,t-1}$$
$$-\begin{cases} 0 & \text{if } t \leq RampUp_p + 1 \\ bProductFamilyDelivered_{p,t-RampUp_p-1} * TotalVehPopulation & \text{if } t > RampUp_p + 1 \end{cases}$$
$$- (1 - bProductFamilyDelivered_{p,t}) * TotalVehPopulation \quad (6.88)$$

Future Program Constraints

- Constraints (6.89) and (6.90) ensure that each future program $\mathcal{F}$ can be activated if and only if at least one of the future systems associated with that program is also activated. These fulfill the first part of the **Future Program Activation** business rule.

$$\forall \mathcal{F}$$
$$\sum_{\mathcal{J} \in FutureProgram_{\mathcal{F}}} bFutureSystem_{\mathcal{J}} \leq NumFutureSystems * bFutureProgram_{\mathcal{F}} \quad (6.89)$$

$$\forall \mathcal{F}$$
$$\sum_{\mathcal{J} \in FutureProgram_{\mathcal{F}}} bFutureSystem_{\mathcal{J}} \geq bFutureProgram_{\mathcal{F}} \quad (6.90)$$

- Constraints (6.91) and (6.92) ensure that each future system $\mathcal{J}$ is activated if and only if at least one brigade of non-future systems is replaced by $\mathcal{J}$ at some time t within the conventional or extended horizon. These partially fulfill the part of the **Future System Fielding** business rule.

$$\forall \mathcal{J}$$
$$\sum_{\substack{i,m,t: \\ (i,\mathcal{J},m) \in FutureTransitions \\ t \leq \mathcal{T}}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq 10,000 * bFutureSystem_{\mathcal{J}} \quad (6.91)$$

$\forall \mathcal{J}$

$$\sum_{\substack{i,m,t:\\(i,\mathcal{J},m)\in FutureTransitions\\t\leq\mathcal{T}}} iNumBrigReplaced_{i,\mathcal{J},m,t} \geq bFutureSystem_{\mathcal{J}} \quad (6.92)$$

- Some future programs may only be activated if every associated future system is fielded. This constraint enforces that behavior of the programs $\mathcal{F}$ where the the user has set the $FieldAllWithinProgram_{\mathcal{F}}$ flag to 1. This fulfills the optional portion of the **Future Program Activation** business rule.

$\forall \mathcal{F}, \mathcal{J} \in FutureProgram_{\mathcal{F}}$ where $FieldAllWithinProgram_{\mathcal{F}} = 1$

$$bFutureProgram_{\mathcal{F}} \leq bFutureSystem_{\mathcal{J}} \quad (6.93)$$

- Constraints (6.94) and (6.95) ensure that once future systems start fielding to a mission, non-future systems are no longer allowed to field to that mission, neither by mission upgrades nor storage swaps. These fulfill the **Future Obviates Present** business rule.

$\forall \mathcal{J}$

$$10,000 * (1 - bFutureSystem_{\mathcal{J}}) \geq \sum_{\substack{i,j,m,t:\\(i,j,m)\in Transitions\\m=FutureMissionMap_{\mathcal{J}}\\FutureSysFirstField_{\mathcal{J}}\leq t\leq\mathcal{T}}} iNumBrigFromStorage_{i,j,m,t} \quad (6.94)$$

$\forall \mathcal{J}$

$$10,000 * (1 - bFutureSystem_{\mathcal{J}}) \geq \sum_{\substack{i,j,m,t:\\(i,j,m)\in Transitions\\m=FutureMissionMap_{\mathcal{J}}\\FutureSysFirstField_{\mathcal{J}}\leq t\leq\mathcal{T}}} iNumBrigInMissionUpgraded_{i,j,m,t} \quad (6.95)$$

- This constraint ensures that if future system $\mathcal{J}$ is activated, then it must fielded according to a fixed schedule given by the input $FutureFieldingProfile_{\mathcal{J},t}$. Note that the optimization can still decided which non-future systems will be replaced first. Together with (6.91) and (6.92), this satisfies the **Future System Fielding** business rule.

$\forall \mathcal{J}, t \leq \mathcal{T}$

$$\sum_{\substack{i,m:\\(i,\mathcal{J},m)\in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} = bFutureSystems_{\mathcal{J}} * FutureFieldingProfile_{\mathcal{J},t} \quad (6.96)$$

- Constraints (6.97) and (6.98) ensure that the number of future systems flowing in must not exceed the number of non-future systems in service that are being replaced. This fulfills the **Outflow Availability** buisness rule in relation to future systems.

$$\forall (i,m) \in Roles$$
$$\sum_{\substack{\mathcal{J},t: \\ (i,\mathcal{J},m) \in FutureTransitions \\ t \leq T}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq NumBrigInMission_{i,m,T} \quad (6.97)$$

$$\forall (i,m) \in Roles, 1 < t \leq T$$
$$\sum_{\substack{\mathcal{J}: \\ (i,\mathcal{J},m) \in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq NumBrigInMission_{i,m,t-1} \quad (6.98)$$

- This constraint ensures that all the future system types that are mandated to field are actually fielded. This helps fulfill the **Future System Fielding** business rule.

$$\forall \mathcal{J} \in FutureMandatedSystems$$
$$bFutureSystem_{\mathcal{J}} + bFutureSystemIndicatorDeficit_{\mathcal{J}} = 1 \quad (6.99)$$

- This constraint ensures that the correct number of groups of future systems that are not fielded are calculated when the indicated future system does not field.

$$\forall \mathcal{J}$$
$$iFutureSystemMandateDeficit_{\mathcal{J}} =$$
$$\sum_{t} FutureFieldingProfile_{\mathcal{J},t} * bFutureSystemIndicatorDeficit_{\mathcal{J}} \quad (6.100)$$

- Constraints (6.101) and (6.102) ensure that the $bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell}$ indicator variable equals 1 if and only if there are ever any transitions to future system $\mathcal{J}$ in mission m and those transitions achieve a density level of $\ell \in UpgDensityLevels_m$ groups. These constraints partially fulfill the **Future Minimum Group Transition Density** business rule.

$$\forall (\mathcal{J}, m, \ell) \in FutureUpgDensityFlags$$
$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m) \in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} \geq$$
$$(bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} + bFutureSystem_{\mathcal{J}} - 1) * \ell \quad (6.101)$$

$$\forall \mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists (\mathcal{J}, m, \ell) \in FutureUpgDensityFlags$$

$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m) \in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq$$

$$\sum_{\substack{\ell: \\ \ell \in UpgDensityLevels_m \\ \ell \neq \max(UpgDensityLevels_m)}} (bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} * \ell)$$

$$+ \sum_{\substack{\ell: \\ \ell \in UpgDensityLevels_m \\ \ell = \max(UpgDensityLevels_m)}} (bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} * BrigPerMission_m) \tag{6.102}$$

- Constraint (6.103) ensures that future system $\mathcal{J}$ in mission m can satisfy at most 1 of the future minimum transition density levels. Note that if future system $\mathcal{J}$ never transitions into mission m at any time, then all three $bFutureTransitionedToDensityLevel$ binaries will be 0. Together with (6.101) and (6.102), this fulfills the **Future Minimum Group Transition Density** business rule.

$$\forall \mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists (\mathcal{J}, m, \ell) \in FutureUpgDensityFlags$$

$$\sum_{\ell \in UpgDensityLevels_m} bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} \leq 1 \tag{6.103}$$

- Constraints (6.104) and (6.105) ensure that the $bFutureHasFinalDensity_{\mathcal{J},m,\ell}$ indicator variable equals 1 if and only if future system $\mathcal{J}$ in mission m has a density level of $\ell \in FinalDensityLevels_m$ groups at time $\mathcal{T}$. These constraints partially fulfill the **Future Minimum Group Final Density** business rule.

$$\forall (\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$

$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m) \in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} \geq$$

$$(bFutureHasFinalDensity_{\mathcal{J},m,\ell} + bFutureSystem_{\mathcal{J}} - 1) * \ell \tag{6.104}$$

$$\forall \mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$

$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m) \in FutureTransitions}} iNumBrigReplaced_{i,\mathcal{J},m,t} \leq$$

$$\sum_{\substack{\ell: \\ \ell \in FinalDensityLevels_m \\ \ell \neq \max(FinalDensityLevels_m)}} (bFutureHasFinalDensity_{\mathcal{J},m,\ell} * \ell)$$

$$+ \sum_{\substack{\ell: \\ \ell \in FinalDensityLevels_m \\ \ell = \max(FinalDensityLevels_m)}} (bFutureHasFinalDensity_{\mathcal{J},m,\ell} * BrigPerMission_m) \tag{6.105}$$

- Constraint (6.106) ensures that future system $\mathcal{J}$ in mission m can satisfy at most 1 of the final transition density levels. Note that if future system $\mathcal{J}$ never transitions into mission m at any time, then all three of the $bFutureHasFinalDensity$ binaries will be 0. Together with (6.104) and (6.105), this fulfills the **Future Minimum Group Final Density** business rule.

$$\forall \mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$

$$\sum_{\ell \in FinalDensityLevels_m} bFutureHasFinalDensity_{\mathcal{J},m,\ell} \leq 1 \tag{6.106}$$

DISTRIBUTION:

1 Roy E. Rice
Teledyne Brown Engineering
300 Sparkman Drive
Huntsville, AL 35805-1912

1	MS 0330	Liliana Shelton, 421
1	MS 0671	Gio K. Kao, 5629
1	MS 1188	Richard O. Griffith, 6130
1	MS 1188	Stephen M. Henry, 6133
1	MS 1188	Matthew J. Hoffman, 6133
1	MS 1188	Craig R. Lawton, 6130
1	MS 1188	Darryl J. Melander, 9529
1	MS 1188	Frank M. Muldoon, 6133
1	MS 1188	Alan Nanco, 6114
1	MS 1188	Bruce M. Thompson, 6133
1	MS 0899	Technical Library, 9536 (electronic copy)

This page intentionally left blank.

Sandia National Laboratories