SANDIA REPORT

SAND # Pending
Unlimited Release (Pending R&A)
Printed September 2017

The Capability Portfolio Analysis Tool (CPAT): A Mixed Integer Linear Programming Formulation for Fleet Modernization Analysis (Version 2.0.2)

Lucas A. Waddell, Frank M. Muldoon,Stephen M. Henry,
Matthew J. Hoffman, April M. Zwerneman, Peter B. Backlund,
Darryl J. Melander, Craig R. Lawton, Roy E. Rice

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

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.

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 (Pending R&A)
Printed September 2017

The Capability Portfolio Analysis Tool (CPAT): A Mixed Integer Linear Programming Formulation for Fleet Modernization Analysis (Version 2.0.2)

Lucas A. Waddell[1], Frank M. Muldoon[1], Stephen M. Henry[1], Matthew J. Hoffman[2], April M. Zwerneman[3], Peter B. Backlund[4], Darryl J. Melander[5], Craig R. Lawton[3], Roy E. Rice[6]

[1] Systems Readiness and Sustainment Technologies, Org 8833
[2] Operations Research and Computational Analysis, Org 8831
[3] Mathematical Analysis and Decision Science, Org 8834
[4] Systems Technology, Org 5837
[5] Software Systems R&D, Org 9365
Sandia National Laboratories
P.O. Box 5800, Albuquerque, New Mexico 87185

[6]Teledyne Brown Engineering
300 Sparkman Drive, Huntsville, AL 35805-1912

Abstract

In order to effectively plan the management and modernization of their large and diverse fleets of vehicles, Program Executive Office Ground Combat Systems (PEO GCS) and Program Executive Office Combat Support and Combat Service Support (PEO CS&CSS) 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 paper, which is an update to the original CPAT formulation document published in 2015 (SAND2015-3487), covers the formulation of important new CPAT features.

Contents

Preface **9**

Nomenclature **11**

1 Business Rules **15**

System Transition Flow 15

Mission Priority Tiers 17

Transition Delays 18

General Scheduling Rules 18

Budgets 20

Product Families 21

Low-Rate Initial Production 23

Coasting Systems 23

Future Programs 24

2 Formulation Indices, Sets and Tuples **25**

Formulation Indices 25

Useful Sets and Tuples 26

Fleet Structure and Flow 26

Optimization Tiers 27

Modernization Scheduling 28

Product Families 29

Coasting Systems 30

Future Programs 31

3 Model Input Parameters **33**

Optimization Tiers and Phases 33

Fleet Structure and Flow 34

Modernization Scheduling 36

Cost & Budgets 38

Future Programs 40

Auxiliary Parameters 40

4 MILP Decision Variables **43**

Non-Negative Integer Variables 43

Binary Variables 45

Continuous "Binary" Variables 46

Non-negative Continuous Variables 47

5 MILP Variable Expressions **49**

Fleet Structure and Flow 49

Low-Rate Initial Production 53

Storage 54

Cost and Budgets 56

Production 66

Product Families 69

Objective Function 70

6 MILP Constraints **75**

Multi-Tier, Multi-Phase Constraints 75

System Flow Conservation Constraints 79

General Scheduling Constraints 83
Budget Constraints 87
Group Density Levels 91
System Production Constraints 93
Product Family Constraints 94
LRIP Constraints 101
Production Smoothing Constraints 103
Coasting Systems Constraints 104
Future Program Constraints 106

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) and Tactical Wheeled Vehicle (TWV) fleets 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 fleets. 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[1], which presents the CPAT methodology from the perspective of an outside analyst not possessing intimate knowledge of the mathematical formulation.

This paper serves as an update to the original Capability Portfolio Analysis Tool (CPAT) formulation document, which was published in 2015[2]. All relevant content from the original publication remains in the interest of completeness. In addition, the following new CPAT features are integrated throughout the formulation:

- *Components* – A "component" is a division of the fleet into separate operational units with similar structure.
- *Batch Sizes* – Batch size refers to the smallest increment of a system that may be purchased.
- *Maximum Time New Systems are in Storage* – The analyst may specify an upper limit on the amount of time a new system may spend in storage before being fielded.

[1]Waddell, Lucas A., Frank M. Muldoon, Darryl J. Melander, Peter B. Backlund, Stephen M. Henry, Matthew J. Hoffman, April M. Zwerneman, Craig R. Lawton, Roy E. Rice, "The CPAT 2.0.2 Domain Model – How CPAT 2.0.2 "Thinks" From an Analyst Perspective," Sandia National Laboratories, Albuquerque, NM SAND# Pending.

[2]Henry, Stephen M., Frank M. Muldoon, Matthew J. Hoffman, Gio K. Kao, Craig R. Lawton, Darryl J. Melander, Liliana Shelton, "The Capability Portfolio Analysis Tool (CPAT): A Mixed Integer Linear Programming Formulation for Fleet Modernization Analysis," Sandia National Laboratories, Albuquerque, NM SAND2015-3487.

- *Minimum Cumulative Delivery* – The analyst may specify a minimum cumulative production for each product family.
- *Product Family Obviation* – The analyst may specify that a product family obviates one or more other product families. I.e., if and when a system from that product family is produced, systems from the obviated product families cannot be produced any longer.
- *Product Family Ratios* – The analyst may specify product family ratios which require that systems produced from a family must be distributed among components according to that ratio.
- *System Coasting* – The analyst may specify some systems as "coastable." A coastable system may be delivered into future time periods at the same rate that it was delivered in the final conventional time period.
- *RDT&E Active Costs* – Similar to procurement active costs, the analyst may specify an amount of RDT&E costs to be incurred in each period that a product family is active. This is in addition to any upfront RDT&E costs a product family may have.
- *Component Earmarks* – Money may be allocated to purchases and upgrades of systems in a specific component (beyond the general combined expenditures budget).
- *Product Family Earmarks* – Money may be allocated to purchases and upgrades of systems in a specific product family (beyond the general combined expenditures budget).
- *Economic Useful Life* – All systems in the fleet are given an economic useful life, which is the maximum age the system is allowed to be before it must be retired from a mission.
- *Arbitrary Phase Ordering* – The analyst may specify the optimization phase ordering to whatever suits their current analysis needs (e.g., schedule, age, yearly budget, horizon budget, performance, and cost).
- *Optional Storage Upgrades* – The analyst has the option to specify for each upgrade whether that upgrade can be done in storage.
- *Hard Limits on Tier Phases* – The analyst has the ability to enforce hard constraints on any of the phase objective metrics.
- *Disallow Instantaneous Cross-Mission Transfers* – Systems retired in any time period cannot be immediately re-fielded in the same time period.
- *Allow Overlap in System Obviations* – The analyst may specify a number of time periods in which both systems (the obviated and the obviating) can be delivered.

Nomenclature

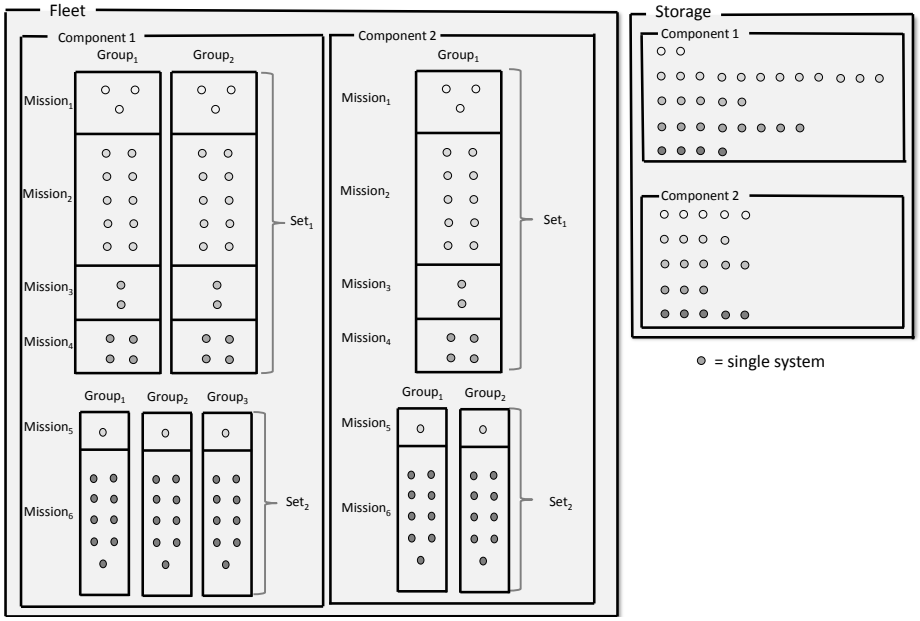

Figure 1. CPAT Fleet and Storage Structure

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 belonging to components). 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.

Component: A division of the fleet into separate operational units with similar structure.

Set: A high-level partition of a component. 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 called "brigades," and we will often use this term interchangeably.

Mission: The partitioning of a group into areas of unique operational responsibility. Every group 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 units 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 in storage are assigned to a component and 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 a component in 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 any component 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 in a specific component. 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 in a component.

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 procurement 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 and coasting systems are made during the extended planning horizon.

This page intentionally left blank.

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 for all components 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 in a component must all be of the same system type. Different groups in the same or different components 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 in a component 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 in a component 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 in each component 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 assigned to components which are also immediately available to begin upgrading or swapping into missions in that component. 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 a particular component in storage, 2) storage upgrades take one system type already in storage and turn it into another type in the same component, and 3) storage swaps take one system type out of a mission from a component and into storage into the same component while taking another type out of storage from the component and sending it to the mission in that component. 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 component in the same time period.

- **No Pre-Usage Upgrades:** Newly purchased systems in storage in any component 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 in a component. However, this ability is optional and may be disallowed by user choice.

- **Optional Storage Upgrades:** Systems may be upgraded in mission or in storage for any component. The user has the option to specify for each upgrade whether that upgrade can be done in storage.

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

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

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

- **Fielding New Systems From Storage:** New systems delivered to storage via purchases or in storage upgrades may only remain in storage for a user defined number of time periods. Additionally, any LRIP systems or systems in initial storage may only remain in storage for the user-specified number of time periods, beginning in the first time period that non-LRIP production is completed.

- **Disallow Instantaneous Cross-Mission Transfers:** Systems retired from a mission in a component in any time period cannot immediately be re-fielded to another mission in a component in the same time period.

Mission Priority Tiers

- **Priority Tiers:** Fleet missions in any component may be partitioned into priority tiers wherein each tier comprises a separate optimization. The modernization of missions in all components 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 in components across multiple tiers; it is not allowed to start up separately for each tier.

- **Tier Phases:** Within each tier, there are six separate optimization phases. The user can determine which phases are executed, and in which order, depending on the analysis question being answered. The optimized value in each phase becomes a constraint on later phases. The phases are:

  - Minimize schedule mandate violations
  - Minimize economic useful life violations
  - Minimize yearly budget violations
  - Minimize horizon (cumulative) budget violations
  - Maximize fleet performance
  - Minimize cumulative combined costs

  If the user selected all six phases in the order above (not typical), they would be executed as follows. The first minimizes schedule violations; the second minimizes economic useful life violations while not allowing schedule violations to increase, the third minimizes yearly budget violations while not allowing either schedule or economic useful life violations to increase; the fourth minimizes horizon budget violations while not allowing schedule, economic useful life, or yearly budget violations to increase; the fifth maximizes fleet performance while not allowing schedule, economic useful life, yearly budget, or horizon budget violations to increase; the sixth minimizes cumulative combined fleet costs while preserving fleet performance and not allowing schedule, economic useful life, yearly budget, or horizon budget violations to increase.

  Mandates, economic useful life constraints and budget violations were all chosen to be minimized penalties rather than hard constraints. This ensures that if such business rules must be broken, the phase ordering prevents trade offs between violations and the user can diagnose the issue. Costs are often minimized after the performance maximization phase to ensure performance is achieved via the most intelligent possible allocation of budget resources and lower tiers, which use the left-over budget from higher tiers, will have the best possible opportunity for modernization.

- **Hard Limits on Tier Phases:** CPAT provides analysts with the ability to enforce hard constraints on any of the phase objective metrics. For example, the analyst may specify that the total number of economic use violations must be less than X,

or that the fleet's cumulative performance must be greater than Y. These limits must be satisfied in all optimization phases. This capability allows the analyst to answer a myriad of questions such as "what is the cheapest fleet modernization plan with less than X economic useful life violations?" and "what is the smallest horizon budget violation required to achieve a fleet with at least Y cumulative performance?"

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

Transition Delays

- **Delay Partitioning:** When upgrading from one system to another (whether in a mission or in storage for any component) 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 must 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 percentage is applied across the fleet and is not specific to systems in each component. This modernization must be performed. Default requirement = 0%.

- **System Mandates:** Missions in each component may mandate that a minimum number of a particular system type be in that mission in that component 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 time period over all components. 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 over all components. 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 across all components. 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 across all components 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, with the option to specify a number of overlap time periods in which both systems can be delivered. This applies to systems in any mission and in any component.

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

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

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

- **Economic Useful Life:** Certain systems may have an upper bound on the number of time periods that they can spend in a mission in any component. Once a system

reaches that specific limit or age then it must be retired from service. This rule does not apply to terminal systems (i.e., systems that cannot be transitioned to any other system).

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 throughout 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.

- **Component Earmarks:** Additional money, above the per-period budget, may be allocated to purchases and upgrades of systems in a specific component in a specific conventional or extended time period. An earmark does not require purchases and upgrades of at least that amount to be spent for that component, but it does not allow any portion of the earmark to be spent for the purchases and upgrades in another component. Default earmarks = 0.

- **Product Family Earmarks:** Additional money, above the per-period budget, may be allocated to active costs, startup costs, LRIP, RDT&E, purchases, and upgrades of systems in a specific product family in a specific conventional or extended time period. An earmark does not require at least that amount to be spent for that product family, but it does not allow any portion of the earmark to be spent for active costs, startup costs, LRIP, or RDT&E of any other product family or for the purchases and upgrades of systems not in the specified product family. When a system is purchased or upgraded in a time period in which it could charge either a product family earmark or a component earmark, the product family earmark is charged. Default earmarks = 0

- **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 procurement and RDT&E 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 in components using a system 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 procurement cost and/or per-period RDT&E 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 in components using a system density weighting method. Default per-period cost = \$0.

- **Family Minimum Per-Period Delivery:** For each product family, there may be a required minimum number of systems to be delivered from that product family in any specific time period. These lower bounds on delivery for each product family must be met. LRIP delivered does not count towards this minimum. Default minimum delivery for each product family and time period is 0.

- **Family Maximum Per-Period Delivery:** 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 Minimum Cumulative Delivery:** For each product family, there may be a lower limit on the cumulative number of member systems that are ever delivered from the family if any systems are delivered from the family. These limits must be respected. All produced LRIP and coasting systems count towards this capacity. Default capacity = 0.

- **Family Maximum Cumulative Delivery:** 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. All produced LRIP and coasting systems 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.

- **Upfront RDT&E Costs:** For each product family, there may be an upfront RDT&E cost such that 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 upfront 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 upfront 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. An upfront RDT&E cost profile cannot be incurred if any of the costs extend into future time periods.

- **Product Family Obviation:** All systems within a product family may be obviated by any system within another product family so that any system from the obviated product family can only be delivered prior to any deliveries of systems from the obviating product family.

- **Product Family Ratios:** For each product family, there may be a ratio defined for each component in which delivery of systems from the product family to each component must be within a set variance of the defined ratios. The delivery ratio must be within a specified time period window. Once a product family starts delivering systems to one component, it must also begin delivering systems to all other components according to the ratio. Once deliveries are complete to one component, the ratio for that component is no longer enforced, but ratios defined between the remaining components must be enforced. After deliveries to a component from a product family with ratios stops, the product family may not restart deliveries to that component in a later time period.

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 and delivered up to 5 years before FRP begins. The optimization chooses how to partition the delivered LRIP systems to components in storage. 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.

Coasting Systems

- **Coasting System Fielding:** Some systems, designated by the user, are allowed to continue delivery to missions in any component in the extended time horizon based on their delivery rates in the last conventional time period. These systems are not considered future systems and have more flexibly as their delivery rates are selected by the optimization. If the optimization chooses to coast a system in a mission in a component, then it coasts that system until the mission has no more groups in which to upgrade to the coasting system. Once delivery of coasting system to a mission in a component ceases, then it cannot be restarted in a later time period. Coasting systems are purchased or upgraded based on how they were acquired in the last conventional time period. By convention, a mission will never be supported by both future systems and systems that are allowed to coast.

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 all of its associated systems must be fielded.

- **Future System Fielding:** When a future system is activated in a component, it must be fielded to its mission in that component according to a fixed, user-defined fielding schedule. Optionally, each future system may be mandated to be fielded in every component. If mandated future systems do not field then the schedule phase indicates infeasibilities.

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

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.
- m denotes missions, which are predefined assignments performed by specific systems with specific performance qualities.
- t denotes time periods (years) in the planning horizon. Note that time periods are partitioned into the conventional and extended time horizons. The extended time horizon allows for the inclusions of future programs.
- T denotes the ending period of the conventional planning horizon.
- $\mathcal{T}$ denotes the ending period of the extended planning horizon, which spans $T < t \leq \mathcal{T}$. Note, if there does not exist an extended time horizon in the model then $\mathcal{T} = T$.
- c denotes components.
- p and q denote product families. Each product family is associated with a start-up cost profile, a per-period procurement cost, a per-period RDT&E cost, a set of upfront RDT&E cost profiles, and a subset of system types. The start-up cost profile and an upfront RDT&E profile must be incurred in order to begin producing systems from the family; the per-period costs must be incurred for every time period that the family is active.
- d denotes the possible number of time periods by which upfront 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 $SyncSetMissions_s$) and systems (given by $SyncSetSys_s$) 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. 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. 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

- $SysMissions$ defines the valid pairings (i, m) where system i can serve mission m.
- $Roles$ defines the valid triples (i, m, c) where system i can serve mission m in component c.
- $InterimRoles$ defines the roles (i, m, c) where system i can serve mission m in component c where i is an interim system (i.e., neither initial nor final) in mission m in component c. Here $InterimRoles \subseteq Roles$ and is used to filter the $bInterimGpCanField_{i,m,c,t}$ variables down to the minimal number needed.
- $MandatedRoles$ defines the roles (i, m, c) for which there exists a requirement that at least a certain number (given by $FinalMandate_{i,m,c}$) of systems i must exist in mission m in component c at the end of the conventional time horizon (time T). Here $MandatedRoles \subseteq Roles$ and is used to filter the $iFinalMandateDeficit_{i,m,c}$ variables down to the minimum number needed.
- $Transitions$ defines the valid quadruples (i, j, m, c) where system i can modernize by some means to system j in mission m in component c.
- $MissionUpg$ defines the valid quadruples (i, j, m, c) where system i can upgrade to system j in mission m in component c. Here, the seed system i is consumed in the creation of system j. Note that $MissionUpg \subseteq Transitions$.
- $StorageUpg$ defines the valid pairs (i, j) where system i can upgrade to system j in storage for any component. As in $MissionUpg$, the seed system i is consumed in the creation of j. Unlike $MissionUpg$, these upgrades happen in storage, not in the context of a mission, and so can define pairs (i, j) not seen in the set $Transitions$.
- $Inflow_{i,m,c}$ defines the set of valid systems j for which $(j, i, m, c) \in Transitions$. These are the systems that can modernize <u>to</u> system i in mission m in component c.

- $Outflow_{i,m,c}$ defines the set of valid systems j for which $(i, j, m, c) \in Transitions$. These are the systems that can be modernized from system i in mission m in component c.
- $PurchasableSys$ defines the set of systems i that are able to be purchased. This is used to filter the $iNumBatchesPurch_{i,c,t}$ variables down to the minimal number needed.
- $DeliverableSys$ defines the set of systems i that are able to be delivered (i.e., finished production). This is used to filter the $bSysDelivered_{i,t}$ variables down to the minimal number needed.
- $FreeInterimUpgSys$ 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" systems in storage.
- $SysWithLripOrInitialStorage$ defines the set of systems i that have either LRIP that is delivered to storage or initial storage inventory. This is used to define those systems that require more careful consideration when implementing the "Fielding New Systems from Storage" business rule.

Optimization Tiers

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

- $fixedSysInMissionUpg$ defines the valid (i, j, m, c, t, N) sextuples where N systems will be upgraded from system i to j in mission m in component c at time $t \leq T$ as determined by preceding tier optimizations. This set will be empty during the first tier.
- $fixedSysFromStorage$ defines the valid (i, j, m, c, t, N) sextuples where N systems undergo a storage swap wherein system i leaves mission m in component c and goes to storage while j leaves storage and enters mission m in component c at time $t \leq T$ as determined by preceding tier optimizations. This set will be empty during the first tier.
- $fixedGpReplaced$ defines the valid $(i, \mathcal{J}, m, c, t, N)$ sextuples where N groups will be upgraded from system i to $\mathcal{J}$ in mission m in component c at time $t \leq \mathcal{T}$ 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, c, N) quadruples where system i in mission m in component c 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.

- $fixedFinalFutureMandateDeficit$ defines the valid $(\mathcal{J}, c, N)$ triplets where system $\mathcal{J}$ in component c has a final deficit of N groups at time $\mathcal{T}$ as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedCoastingLevel$ defines the valid (i, m, c, N) quadruples where system i in mission m in component c is coasting at a level of N groups per year in future time periods as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedGpCoastingPurch$ defines the valid (i, j, m, c, t, N) sextuples where N groups of system j are purchased to replace system i in mission m in component c in time $T < t \leq \mathcal{T}$ as determined by preceding tier optimizations. This set will be empty during the first tier.

- $fixedGpCoastingUpg$ defines the valid (i, j, m, c, t, N) sextuples where N groups of system i are upgraded to system j in mission m in component c in time $T < t \leq \mathcal{T}$ as determined by preceding tier optimizations. This set will be empty during the first tier.

Modernization Scheduling

- $ComponentMissionSuccessions$ defines the valid component-mission pairings (c_1, m_1, c_2, m_2) where mission m_1 in component c_1 must completely modernize and finish fielding before m_2 in component c_2 can begin fielding (m_1 in c_1 is said to precede m_2 in c_2).

- $SuccessorComponentMissions$ defines the set of component-mission pairs (c, m) which are preceded by some other component-mission pair as defined in the $ComponentMissionSuccessions$ set.

- $SysObviations$ defines the valid system pairings and overlaps (i, j, o) where system j can only be produced before system i is produced (i obviates j). If an overlap o is specified, both systems can be produced for an overlap of o time periods, starting in the first time period in which system i is produced.

- $SyncSetComponents_s$ defines the component of the synchronization set s.

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

- $SyncSetSys_s$ 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 across all components 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 across all components 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.

- $StorageUpgUsePriorities$ 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.

- $StorageUpgBeforePurch$ 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.

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

- $PfWithAnyActive$ defines product families p for which a per-period procurement active cost and/or a per-period RDT&E active cost exists. This filters the $bPfActive_{p,t}$ variables down to the minimal number needed.

- $PfWithProcureActive$ defines product families p for which a per-period procurement active cost exists.

- $PfWithRdteActive$ defines product families p for which a per-period RDT&E active cost exists.

- $PfWithRdte$ defines product families p with upfront RDT&E cost profile(s) and/or a per-period RDT&E active cost.

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

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

- $PfWithStartup$ defines product families p for which a start-up cost profile exists. This filters the $bPfStartup_{p,t}$ 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 minimum horizon production limit, a maximum horizon production limit, a delivery variance limit for production smoothing, or the restriction that production gaps are

not allowed. This filters the $bPfDelivered_{p,t}$ variables down to the minimal number needed.

- $PfWithRatios$ defines the product families p for which fielding ratios exist between the components. This filters the variables $bPfFirstYearFielding_{p,t}$ and the $bPfLastYearFielding_{p,c,t}$ down to the minimal number needed.

- $PfWithEarmarks_t$ defines the set of product families p in time period t with a non-zero earmark budget.

- $SysInPfWithEarmarks_t$ defines the set of systems that are in product families with non-zero earmark budgets in time t.

- $PfWithLrip$ defines product families p which has an LRIP profile. This filters the $bPfFrpStarted_{p,t}$ 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.

- $PfObviations$ defines the valid product family pairings (p, q) where any system from product family q can only be produced before any systems are produced from product family p (p obviates q).

Coasting Systems

- $CoastableSys$ defines the set of systems i that can continue delivery at a constant rate after the last conventional time period. Coasting systems continue to be delivered at the same rate as their delivery rate in the last conventional time frame. Coasting systems should not be confused with future systems.

- $CoastableRoles$ defines the valid (i, m, c) triples where system i in mission m in component c is allowed to coast in future time periods.

- $CoastablePurchTransitions$ defines the valid (i, j, m, c) quadruples where system j is purchased to replace system i in mission m in component c.

- $CoastableUpgTransitions$ defines the valid (i, j, m, c) quadruples where system i is in mission upgraded to system j in mission m in component c.

Future Programs

- $FutureProgram_{\mathcal{F}}$ defines the set of future system types belonging to program $\mathcal{F}$.
- $FutureTransitions$ defines the valid quadruples $(i, \mathcal{J}, m, c)$ where system i can be replaced by future system $\mathcal{J}$ in mission m in component c.
- $FutureTransitionPurch$ is a subset of $FutureTransitions$ and defines the valid transition $(i, \mathcal{J}, m, c)$ where system i can be replaced by future system $\mathcal{J}$ in mission m in component c and system i returns to storage.
- $FutureMissionMap_{\mathcal{J}}$ defines the mission to which future system $\mathcal{J}$ is fielded.
- $MandatedFutureSys$ defines the set of future system types $\mathcal{J}$ 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 in all components 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.

- $MissionTier_m$ 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.

- $SchedulePhase$, $AgePhase$, $YearlyBudgetPhase$, $HorizonBudgetPhase$, $PerformancePhase$, and $CostPhase$ are binary parameters indicating which of the six 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.

- $TierAgeOveragesBound$ gives the upper limit of the economic useful life or age violation amount allowed during phases subsequent to the age phase within a tier. This limit equals the minimum system-year age violations discovered during the age phase.

- $YearlyBudgetOverrunBound$ gives the upper limit on the yearly budget overage amount allowed during phases subsequent to the yearly budget phase. This limit

equals the minimum yearly budget overrun amount discovered in the yearly budget phase.

- $HorizonBudgetOverrunBound$ gives the upper limit on the horizon budget overage amount allowed during phases subsequent to the horizon budget phase. This limit equals the minimum horizon budget overrun amount discovered in the horizon 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.
- $MaximumCost$ gives the largest acceptable value for all combined costs included in the cost phase expenses during phases subsequent to the cost phase. This upper bound equals the minimum combined fleet cost for only costs included in the cost phase found during the cost phase.
- $ScheduleViolationLimit$ is an optional hard limit (upper bound) that the user can place on the total number of schedule violations. This limit must be obeyed in all optimization phases.
- $AgeViolationLimit$ is an optional hard limit (upper bound) that the user can place on the total number of economic useful life violations. This limit must be obeyed in all optimization phases.
- $YearlyBudgetOverrunLimit$ is an optional hard limit (upper bound) that the user can place on the yearly budget overages. This limit must be obeyed in all optimization phases.
- $TotalBudgetOverrunLimit$ is an optional hard limit (upper bound) that the user can place on the horizon budget overruns. This limit must be obeyed in all optimization phases.
- $PerformanceLimit$ is an optional hard limit (lower bound) that the user can place on the cumulative fleet performance. This limit must be obeyed in all optimization phases.
- $ScheduleViolationLimit$ is an optional hard limit (upper bound) that the user can place on the value for all combined costs included in the cost phase expenses. This limit must be obeyed in all optimization phases.

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.

- $SysPerComponentMission_{c,m}$ gives the total number of systems needed for mission m in component c. Note that this value must equal the total number of systems in the initial inventory for mission m in component c.

- $GpPerComponentMission_{c,m}$ gives the total number of groups needed for mission m in component c.

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

- $maxMissionRequirement_i$ gives the maximum number of systems per group across all missions that system i supports.

- $InitialSysInMission_{i,m,c}$ gives the initial number of systems i in mission m in component c. 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 $SysPerComponentMission_{c,m}$ parameter.

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

- $InitialSysInStorage_i$ 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.

- $InitialRoleAge_{i,m,c}$ gives the initial age of the systems i in mission m in component c. This parameter describes the average age of all the initial systems in mission for all components at the beginning of the time horizon.

- $maxTimeAllowedInStorage$ is the maximum number of time periods a newly purchased or in storage upgraded system is allowed to stay in storage before being fielded to a mission in a component. This time limit counts the year the system was delivered to storage because this is the first year it could be fielded.

Modernization Scheduling

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

- $FinalMandate_{i,m,c}$ gives the minimum number of systems i in mission m in component c 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.

- $GpTransitionLimitPerTime_m$ 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.

- $GpTransitionLimitTotal_m$ 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.

- $PfMsr_p$ 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.

- $PfDeliveryMin_{p,t}$ defines the lower limit on the number of systems from product family p that must be delivered in time period t.

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

- $PfTotalDeliveryMin_p$ gives the lower limit on the cumulative number of systems from product family p that must be delivered if any production in the product family p occurs.

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

- $PfComponentRatios_{p,c}$ defines the ratio on the number of systems delivered from product family p to component c over the $PfRatioWindow_p$ number of time periods.

- $PfRatioWindow_p$ gives the number of time periods for product family p in which the delivery for each component must be within the specified ratio variance.

- $PfRatioVariance_p$ gives the maximize allowable deviation above or below the desired number of systems delivered to components based on the ratios for product family p. For example, if $PfComponentRatios_{p,1} = 2$, $PfComponentRatios_{p,2} = 3$, $PfRatioWindow_p = 1$, and $PfRatioVariance_p = 0.1$, then if product family p delivers 100 systems to component 1 in time period 5, then it must also deliver $\frac{3}{2} * 100 \pm 0.1 * \frac{3}{2} * 100$ to component 2 in time period 5.

- $PfAllowGaps_p$ is a binary flag that takes on value 1 to indicate that delivery 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.

- $LripSeedSys_{p,i}$ 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 $LripSeedSys_{p,i}$) is removed from storage in producing LRIP systems i in product family p.

- $SysUsefulLife_i$ gives the economic useful life of system i in the fleet. Once a system in the fleet reaches this age it must be retired from the fleet.

Cost & Budgets

- $PfStartupCostSchedule_{p,t}$ 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 an FRP system in the family first becomes active, $t = 1$ represents the year after an FRP system in the family first becomes active, and $t = -1$ represents the year before an FRP system in the family first becomes active. All are valid time periods to incur start-up costs.

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

- $PfRdteActiveCost_p$ gives the recurring RDT&E 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 $LripPreCost_{p,i}$ gives the amount charged one period prior to $LripCost_{p,i}$; otherwise $LripPreCost_{p,i} = 0$.

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

- $RdteCostProfiles_{p,d,t}$ gives the upfront 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.

- $HasRdteInFutureTimePeriod_{p,d}$ indicates whether or not the upfront RDT&E cost profile associated with product family p and a delay of d time periods includes any costs that are incurred in a future time period.

- $ProcureBudget_t$ 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 start-up and procurement active 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. This includes both upfront RDT&E costs and RDT&E active costs. 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.

- $ComponentEarmark_{c,t}$ gives a specified allotment of money at time $t \leq \mathcal{T}$ reserved for only purchases and upgrades of systems in component c. Once this allotment of money is exhausted in time period t, component c may use money available in either the $ProcureBudget_t$ or the $CombinedBudget_t$ (if $CombinedBudget_t$ includes procurement expenses). No component may spend, in any way, the earmarks for another component.

- $PfEarmark_{p,t}$ gives a specified allotment of money at time $t \leq \mathcal{T}$ reserved for only purchases and upgrades of systems assigned to product family p or for the active costs, startup costs, RDT&E, or LRIP associated with product family p. Once this allotment of money is exhausted in time period t, product family p may use money available in either the $ProcureBudget_t$, $RdteBudget_t$, or the $CombinedBudget_t$ (if $CombinedBudget_t$ includes procurement and/or RDT&E expenses). No product family may spend, in any way, the earmarks for another product family.

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

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

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

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

Future Programs

- $FutureSysFieldingProfile_{\mathcal{J},c,t}$ gives the number of groups of future systems of type $\mathcal{J}$ in component c that must be fielded at time $t \leq \mathcal{T}$, if the future system is activated.
- $FutureSysFirstYearFielding_{\mathcal{J}}$ gives the time period during which future system $\mathcal{J}$ first fields to any component, according to the schedule dictated by $FutureSysFieldingProfile_{\mathcal{J},c,t}$.
- $FutureProcureBudget_t$, $FutureOSBudget_t$, $FutureRdteBudget_t$, and $FutureCombinedBudget_t$ give the procurement, O&S, RDT&E, and Combined budgets, respectively, for time periods in the extended time horizon (i.e., $T < t \leq \mathcal{T}$).
- $FutureProgramRdteCostSchedule_{\mathcal{F},t}$, $FutureProgramStartupCostSchedule_{\mathcal{F},t}$, and $FutureProgramActiveCostSchedule_{\mathcal{F},t}$ give the cost of future program $\mathcal{F}$ (if activated) at time t in the categories of RDT&E, startup, and procurement active cost, respectively.
- $FutureTransitionCost_{i,\mathcal{J},m,c}$ gives the cost of transitioning 1 system from type i to future system type $\mathcal{J}$ in mission m in component c.
- $FutureSysOSCost_{\mathcal{J}}$ gives the cost of operating and supporting future system $\mathcal{J}$ for one time period.
- $FutureTransitionLongLead_{i,\mathcal{J},m,c}$ gives the fraction of the transition cost from system i to future system $\mathcal{J}$ in mission m in component c that is incurred a year earlier than normal.
- $FutureTransitionDelay_{i,\mathcal{J},m,c}$ 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 in component c.
- $FutureSysLripProfile_{\mathcal{F},c,t}$ gives the number of LRIP systems that program $\mathcal{F}$ produces in component c in time t.
- $FutureSysLripDelay_{\mathcal{J}} = FutureTransitionDelay_{i,\mathcal{J},m,c}$ for that quadruple $(i, \mathcal{J}, m, c)$ having the largest $FutureTransitionCost_{i,\mathcal{J},m,c}$.
- $FutureSysLripLongLead_{\mathcal{J}} = FutureTransitionLongLead_{i,\mathcal{J},m,c}$ for that quadruple $(i, \mathcal{J}, m, c)$ having the largest $FutureTransitionCost_{i,\mathcal{J},m,c}$.
- $FutureSysLripCost_{\mathcal{J}} = FutureTransitionCost_{i,\mathcal{J},m,c}$ for that quadruple $(i, \mathcal{J}, m, c)$ having the largest $FutureTransitionCost_{i,\mathcal{J},m,c}$.

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 $T + 1$.
- $maxGpTransitionLimit$ gives $\max_m GpTransitionLimitPerTime_m$.
- $TotalSysPopulation$ gives the total number of systems in the fleet.
- $NumFutureSys$ is the total number of types of future systems.
- $NumStorageUpg$ is the total number of supported in storage upgrades.
- $NumComponents$ is the total number of components in the fleet.

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 $iModernizedDeficit_{i,m,t}$, $iFinalMandateDeficit_{i,m,c}$, $iFutureSysMandateDeficit_{\mathcal{J},c}$, $iSysYearAgeOverages_{i,m,c,t}$, and $iPfDeliveryDeficits_{p,t}$ 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

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

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

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

- $iNumInStorageUpg_{i,j,c,t}$ denotes the number of systems of type i upgraded in storage to type j for component c at time t. This is defined for all $(i, j) \in StorageUpg$, for all c, 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 $(i, m) \in SysMissions$ and for all $t \leq \mathcal{T}$.

- $iFinalMandateDeficit_{i,m,c}$ denotes how far system i in mission m in component c is below the end-of-conventional-horizon system mandate (given by $FinalMandate_{i,m,c}$). 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 groups 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 $(i, m, c) \in MandatedRoles$.

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

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

- $iSysYearAgeOverages_{i,m,c,t}$ denotes how many system-years of system i in mission m in component c is over its economic useful life. This variable is defined for all $(i, m, c) \in Roles$ and for all $t \leq \mathcal{T}$.

- $iPfDeliveryDeficit_{p,t}$ denotes how many systems below the minimum delivery requirement (given by $PfDeliveryMin_{p,t}$) for product family p at time t. This variable is defined for all p and for all $t \leq T$.

- $iCoastingLevel_{i,m,c}$ denotes how many groups of system i in mission m in component c will continue to be delivered in each year after the end of the conventional time horizon until mission m in component c only contains systems of type i. This variable is defined for all $(i, m, c) \in Roles$.

- $iNumGpCoastingPurch_{i,j,m,c,t}$ denotes how many groups of system type j are purchased to replaced system type j in mission m in component c at time t. This is defined for all $(i, j, m, c) \in CoastablePurchTransitions$ and for all $T < t \leq \mathcal{T}$.

- $iNumGpCoastingUpg_{i,j,m,c,t}$ denotes how many groups of system type i are in mission upgraded to system type j in mission m in component c at time t. This is defined for all $(i, j, m, c) \in CoastableUpgTransitions$ and for all $T < t \leq \mathcal{T}$.

- $iInitialInStorage_{i,c}$ denotes the number of systems of type i that are assigned to component c in the initial storage inventory. This is defined for all i and for all c.

- $iLripProduced_{c,p,i,t}$ denotes the number of systems of type i produced from LRIP for product family p in component c at time t. This is defined for all c, for all $(p, i) \in LripProfiles$, and for all $t \leq T$.

- $iLripDelivered_{c,p,i,t}$ denotes the number of systems of type i delivered from LRIP for product family p in component c at time t. This is defined for all c, for all $(p, i) \in LripProfiles$, and for all $t \leq T$.

- $iLripSeedsConsumed_{i,c,t}$ denotes the number of systems of type i that are consumed as seeds for LRIP in component c at time t. This is defined for all i, for all c, and for all $t \leq T$.

Binary Variables

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

- $bSysDeliveredComponent_{i,c,t}$ denotes whether at least 1 non-LRIP system of type i is delivered (i.e., completes production) to component c at time t. This is defined for all $i \in DeliverableSys$, for all c, and for all $t \leq T$.

- $bSysHasDeliveredComponent_{i,c,t}$ denotes whether at least 1 non-LRIP system of type i has been delivered (i.e., completed production) to component c at or before time t. This is defined for all $i \in DeliverableSys$, for all c, and for all $t \leq T$.

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

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

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

- $bLripSysBaseYear_{p,i,t}$ 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.

- $bPfActive_{p,t}$ denotes whether product family p is active at time t. This is needed only for families having either a procurement or RDT&E per-period active cost. Hence, it is defined for all $p \in PfWithAnyActive$ and for all $t \leq \mathcal{T}$.

- $bPfStartup_{p,t}$ denotes whether time $t \leq T$ is the first year that an FRP in product family p first becomes active.

- $bPfEnforceRatio_{p,c,t}$ denotes whether product family p must enforce its fielding ratio for component c at time t. This is defined for all $p \in PfWithRatios$, for all c, and for all $t \leq T$.

- $bPfFirstYearFielding_{p,t}$ denotes the year t in which product family p first fields from any component. This is defined for all $p \in PfWithRatios$ and for all $t \leq T$.

- $bPfLastYearFielding_{p,c,t}$ denotes the year t in which product family p last fields from component c. This is defined for all $p \in PfWithRatios$, for all c, and for all $t \leq T$.

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

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

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

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

- $bFutureSysComponent_{\mathcal{J},c}$ denotes whether future system $\mathcal{J}$ in component c is activated.

- $bFutureSysDeficit_{\mathcal{J},c}$ denotes that a mandated future system $\mathcal{J}$ in component c does not field.

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

- $bIsCoasting_{i,m,c,t}$ denotes whether groups of system type i in mission m in component c are being delivered in time period t as coasting systems. This is defined for all $(i, m, c) \in CoastableRoles$ and for all $T < t \leq \mathcal{T}$.

Continuous "Binary" Variables

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

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

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

- $bPfEverDelivered_p$ denotes whether any system from family p has been delivered.

- $bPfFrpStarted_{p,t}$ 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 $RdteCostProfiles_{p,d,t}$ is used. If product family p is not used, then $bRdteDelay_{p,d} = 0$ for all d. Defined for all $p \in PfWithRdteDelays$.

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 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 $T < t \leq \mathcal{T}$
- $fTotalProcureBudgetOverrun$
- $fTotalOSBudgetOverrun$
- $fTotalRdteBudgetOverrun$
- $fTotalCombinedBudgetOverrun$

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

- $fMedianDeliveryLevel_p$

The following variable is used to represent the amount of component earmark money spent for purchases and upgrades in component c in time t.

- $fComponentEarmarkSpent_{c,t}$ for all c and for all $t \leq \mathcal{T}$

The following variable is used to represent the amount of product family earmark money spent for purchases and upgrades of systems belonging to product family p or the procurement active costs, startup costs, or LRIP costs associated with product family p in time t.

- $fPfEarmarkSpentProduction_{p,t}$ for all p and for all $t \leq \mathcal{T}$

The following variable is used to represent the amount of product family earmark money spent for upfront or active RDT&E for product family p in time t.

- $fPfEarmarkSpentRdte_{p,t}$ for all p and for all $t \leq \mathcal{T}$

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 one to two dozen additional variable expressions to the ones documented below. However, these additional expressions serve an auxiliary role (mainly for aiding output of optimization data) and do not affect the mathematical structure of the formulation itself. For that reason, we have opted not to include these auxiliary expressions in this document.

Fleet Structure and Flow

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

$$\begin{aligned}&\forall(i,j,m,c) \in MissionUpg, t \leq T\\ &\quad NumSysInMissionUpg_{i,j,m,c,t} = \\ &\qquad iNumGpInMissionUpg_{i,j,m,c,t} * SysPerGp_m\end{aligned} \tag{5.1}$$

- $NumSysFromStorage_{i,j,m,c,t}$ denotes the number of systems i that are swapped out for system j in mission m in component c 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 $iNumGpFromStorage_{i,j,m,c,t}$.

$$\begin{aligned}&\forall(i,j,m,c) \in Transitions, t \leq T\\ &\quad NumSysFromStorage_{i,j,m,c,t} = \\ &\qquad iNumGpFromStorage_{i,j,m,c,t} * SysPerGp_m\end{aligned} \tag{5.2}$$

- $NumGpInMission_{i,m,c,t}$ denotes the number of groups of system i performing in mis-

sion m in component c at time t.

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

$$\begin{aligned}
NumGpInMission_{i,m,c,t} = {} & InitialGpInMission_{i,m,c} \\
& + \sum_{\substack{j,t^*: \\ j \in Inflow_{i,m,c} \\ t^* \leq \min\{t,T\}}} iNumGpFromStorage_{j,i,m,c,t^*} \\
& - \sum_{\substack{j,t^*: \\ j \in Outflow_{i,m,c} \\ t^* \leq \min\{t,T\}}} iNumGpFromStorage_{i,j,m,c,t^*} \\
& + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in MissionUpg \\ t^* \leq \min\{t,T\}}} iNumGpInMissionUpg_{j,i,m,c,t^*} \\
& - \sum_{\substack{j,t^*: \\ (i,j,m,c) \in MissionUpg \\ t^* \leq \min\{t,T\}}} iNumGpInMissionUpg_{i,j,m,c,t^*} \\
& + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in CoastablePurchTransitions \\ T < t^* \leq t}} iNumGpCoastingPurch_{j,i,m,c,t^*} \\
& - \sum_{\substack{j,t^*: \\ (i,j,m,c) \in CoastablePurchTransitions \\ T < t^* \leq t}} iNumGpCoastingPurch_{i,j,m,c,t^*} \\
& + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in CoastableUpgTransitions \\ T < t^* \leq t}} iNumGpCoastingUpg_{j,i,m,c,t^*} \\
& - \sum_{\substack{j,t^*: \\ (i,j,m,c) \in CoastableUpgTransitions \\ T < t^* \leq t}} iNumGpCoastingUpg_{i,j,m,c,t^*} \\
& - \sum_{\substack{\mathcal{J},t^*: \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ t^* \leq t}} iNumGpReplaced_{i,\mathcal{J},m,c,t^*}
\end{aligned} \tag{5.3}$$

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

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

$$NumSysInMission_{i,m,c,t} = NumGpInMission_{i,m,c,t} * SysPerGp_m \tag{5.4}$$

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

upgrade) in time t.

$$\begin{aligned}
&\forall (i,m,c) \in Roles, t \leq T \\
&\quad NumSysInMissionExchangeable_{i,m,c,t} = \\
&\qquad NumSysInMission_{i,m,c,t} \\
&\qquad - \sum_{j:\ (j,i,m,c) \in MissionUpg} NumSysInMissionUpg_{j,i,m,c,t} \\
&\qquad - \sum_{j \in Inflow_{i,m,c}} NumSysFromStorage_{j,i,m,c,t} \\
&\qquad - \sum_{\substack{j,t^*: \\ (i,j,m,c) \in MissionUpg \\ t < t^* \leq t + UpgProdDelay_{i,j}}} NumSysInMissionUpg_{i,j,m,c,t^*}
\end{aligned} \tag{5.5}$$

- $TransitionedToRole_{i,m}$ 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.

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

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

$$\begin{aligned}
&\forall (i,j,m,c) \in Transitions, t \leq T \\
&\quad NumGpTransit_{i,j,m,c,t} = iNumGpFromStorage_{i,j,m,c,t} \\
&\qquad + iNumGpInMissionUpg_{i,j,m,c,t}
\end{aligned} \tag{5.7}$$

- $SysFieldedToRoleSoFar_{i,m,c,t}$ denotes the total number of systems i that are transi-

tioned into mission m in component c during time periods up to and including t.

$$\begin{aligned}
&\forall (i,m,c) \in Roles, t \leq \mathcal{T} \\
&\quad SysFieldedToRoleSoFar_{i,m,c,t} = \\
&\qquad \sum_{\substack{j,t^*: \\ j \in Inflow_{i,m,c} \\ t^* \leq \min\{t,T\}}} NumSysFromStorage_{j,i,m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in MissionUpg \\ t^* \leq \min\{t,T\}}} NumSysInMissionUpg_{j,i,m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in CoastablePurchTransitions \\ T < t^* \leq t}} iNumGpCoastingPurch_{j,i,m,c,t^*} * SysPerGp_m \\
&\qquad + \sum_{\substack{j,t^*: \\ (j,i,m,c) \in CoastableUpgTransitions \\ T < t^* \leq t}} iNumGpCoastingUpg_{j,i,m,c,t^*} * SysPerGp_m
\end{aligned} \tag{5.8}$$

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

$$\begin{aligned}
&\forall (i,m,c) \in Roles, t \leq \mathcal{T} \\
&\quad GpRetiredFromRoleSoFar_{i,m,c,t} = \\
&\qquad \sum_{\substack{j,t^*: \\ j \in Outflow_{i,m,c} \\ t^* \leq \min\{t,T\}}} iNumGpFromStorage_{i,j,m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (i,j,m,c) \in MissionUpg \\ t^* \leq \min\{t,T\}}} iNumGpInMissionUpg_{i,j,m,c,t^*} \\
&\qquad + \sum_{\substack{\mathcal{J},t^* \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ t^* \leq t}} iNumGpReplaced_{i,\mathcal{J},m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (i,j,m,c) \in CoastablePurchTransitions \\ T < t^* \leq t}} iNumGpCoastingPurch_{i,j,m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (i,j,m,c) \in CoastableUpgTransitions \\ T < t^* \leq t}} iNumGpCoastingUpg_{i,j,m,c,t^*}
\end{aligned} \tag{5.9}$$

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

$$\forall (i,m,c) \in Roles, t \leq \mathcal{T}$$
$$SysRetiredFromRoleSoFar_{i,m,c,t} = GpRetiredFromRoleSoFar_{i,m,c,t} * SysPerGp_m \tag{5.10}$$

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

$$\forall (i,\mathcal{J},m,c) \in FutureTransitions, t \leq \mathcal{T}$$
$$NumSysReplaced_{i,\mathcal{J},m,c,t} = iNumGpReplaced_{i,\mathcal{J},m,c,t} * SysPerGp_m \tag{5.11}$$

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.

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$LripDelivered_{p,i,t} = \sum_{\substack{t^* \in LripYears \\ t+t^* \leq T}} LripPreDelivery_{p,i,t^*} * bLripSysBaseYear_{p,i,t+t^*} \tag{5.12}$$

- $LripProduced_{p,i,t}$ denotes the number of LRIP systems i in product family p that finish production in year t. Note that depending on the LRIP delivery profile, some of these produced systems might not be put into storage.

$$\forall (p,i) \in LripProfiles, t$$
$$LripProduced_{p,i,t} = \sum_{\substack{t^* \in LripYears \\ t+t^* \leq T}} LripPreProduction_{p,i,t^*} * bLripSysBaseYear_{p,i,t+t^*} \tag{5.13}$$

- $LripSeedsConsumed_{i,t}$ 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.

$$\forall i, t \leq T$$
$$LripSeedsConsumed_{i,t} = \sum_{\substack{p,j,t^*: \\ (p,j) \in LripProfiles \\ t^* \in LripYears \\ LripSeedSys_{p,j}=i \\ LripConsumesSeeds_{p,j}=1 \\ t+t^*+LripDelay_{p,j} \leq T}} LripPreProduction_{p,j,t^*} * bLripSysBaseYear_{p,j,t+t^*+LripDelay_{p,j}} \tag{5.14}$$

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

$$\forall i, t \leq T$$
$$NumLripSysActive_{i,t} = \sum_{\substack{p,t^*,t^{**}: \\ p \in SysPfMembership_i \\ t^* \in LripYears \\ t^{**}-t^*-LripDelay_{p,i} \leq t < t^{**}-t^*+\beta_{p,i}}} LripPreProduction_{p,i,t^*} * bLripSysBaseYear_{p,i,t^{**}} \tag{5.15}$$

Storage

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

$$\forall i \in PurchasableSys, c, t \leq T$$
$$NumSysPurch_{i,c,t} = PurchBatchSize_i * iNumBatchesPurch_{i,c,t} \tag{5.16}$$

- $NumSysInStorage_{i,c,t}$ denotes the number of systems i assigned to component c that are currently in storage at time t. This is calculated by adding what you start with 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.

$$
\begin{aligned}
&\forall i, c, t \leq T \\
&\quad NumSysInStorage_{i,c,t} = iInitialInStorage_{i,c} \\
&\qquad + \sum_{t^* \leq t} NumSysPurch_{i,c,t^*} \\
&\qquad + \sum_{\substack{j,m,t^*: \\ (i,j,m,c) \in Transitions \\ t^* \leq t}} NumSysFromStorage_{i,j,m,c,t^*} \\
&\qquad + \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpg \\ t^* \leq t}} iNumInStorageUpg_{j,i,c,t^*} \\
&\qquad + \sum_{\substack{p,t^*: \\ (p,i) \in LripProfiles \\ t^* \leq t}} iLripDelivered_{c,p,i,t^*} \\
&\qquad - \sum_{\substack{j,m,t^*: \\ (j,i,m,c) \in Transitions \\ t^* \leq t}} NumSysFromStorage_{j,i,m,c,t^*} \\
&\qquad - \sum_{\substack{j,t^*: \\ (i,j) \in StorageUpg \\ t^* \leq t}} iNumInStorageUpg_{i,j,c,t^*} \\
&\qquad - \sum_{t^* \leq t} iLripSeedsConsumed_{i,c,t^*} \\
&\qquad + \sum_{\substack{\mathcal{J},m,t^*: \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ t^* \leq t}} NumSysReplaced_{i,\mathcal{J},m,c,t^*}
\end{aligned}
\tag{5.17}
$$

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

$$
\begin{aligned}
&\forall i, c, t \leq T \\
&\quad NumSysInStorageExchangeable_{i,c,t} = NumSysInStorage_{i,c,t} \\
&\qquad - \sum_{\substack{j,t^*: \\ (i,j) \in StorageUpg \\ t < t^* \leq t + UpgProdDelay_{i,j}}} iNumInStorageUpg_{i,j,c,t^*}
\end{aligned}
\tag{5.18}
$$

Cost and Budgets

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

$$\forall p \in PfWithStartup, t \leq T$$
$$PfStartupCost_{p,t} = \sum_{t^* \leq T} bPfStartup_{p,t^*} * PfStartupCostSchedule_{p,t-t^*} \tag{5.19}$$

- $ComponentPurchAndUpgExpense_{c,t}$ denotes the amount spent solely on purchases, in mission upgrades, in storage upgrades, and delivered LRIP of non-future systems for component c in time t within the conventional time horizon.

$$\forall c, t \leq T$$
$$\begin{aligned} ComponentPurchAndUpgExpense_{c,t} = & \sum_{i \in PurchasableSys} NumSysPurch_{i,c,t+PurchDelay_i+1} * PrePurchaseCost_i \\ & + \sum_{i \in PurchasableSys} NumSysPurch_{i,c,t+PurchDelay_i} * PurchCost_i \\ & + \sum_{\substack{i,j,m: \\ (i,j,m,c) \in MissionUpg}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\ & + \sum_{\substack{i,j,m: \\ (i,j,m,c) \in MissionUpg}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\ & + \sum_{(i,j) \in StorageUpg} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\ & + \sum_{(i,j) \in StorageUpg} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\ & + \sum_{\substack{p,i,: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i}+1 \leq T}} LripPreCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}+1} \\ & + \sum_{\substack{p,i,: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i} \leq T}} LripCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}} \end{aligned} \tag{5.20}$$

- $ComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t}$ denotes the amount spent solely on purchases, in mission upgrades, in storage upgrades, and delivered LRIP of non-future systems not assigned to a product family with an earmark in time t in the

conventional time horizon for component c.

$\forall c, t \leq T$

$$\begin{aligned}
&ComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t} = \\
&\quad \sum_{\substack{i \in PurchasableSys: \\ i \notin SysInPfWithEarmark_t}} NumSysPurch_{i,c,t+PurchDelay_i+1} * PrePurchaseCost_i \\
&\quad + \sum_{\substack{i \in PurchasableSys: \\ i \notin SysInPfWithEarmark_t}} NumSysPurch_{i,c,t+PurchDelay_i} * PurchCost_i \\
&\quad + \sum_{\substack{i,j,m: \\ (i,j,m,c) \in MissionUpg \\ j \notin SysInPfWithEarmark_t}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\
&\quad + \sum_{\substack{i,j,m: \\ (i,j,m,c) \in MissionUpg \\ j \notin SysInPfWithEarmark_t}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\
&\quad + \sum_{\substack{(i,j) \in StorageUpg: \\ j \notin SysInPfWithEarmark_t}} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j} \\
&\quad + \sum_{\substack{(i,j) \in StorageUpg: \\ j \notin SysInPfWithEarmark_t}} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}} * UpgCost_{i,j} \\
&\quad + \sum_{\substack{p,i,: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i}+1 \leq T \\ i \notin SysInPfWithEarmark_t}} LripPreCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}+1} \\
&\quad + \sum_{\substack{p,i,: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i} \leq T \\ i \notin SysInPfWithEarmark_t}} LripCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}} \qquad (5.21)
\end{aligned}$$

- $PurchAndUpgExpense_t$ denotes the amount spent solely on purchases, in mission upgrades, in storage upgrades, and delivered LRIP of non-future systems for all components in time t within the conventional time horizon.

$\forall t \leq T$

$$PurchAndUpgExpense_t = \sum_c ComponentPurchAndUpgExpense_{c,t} \qquad (5.22)$$

- $PfPurchAndUpgExpenses_{p,t}$ denotes the amount spent solely on purchases, in mission upgrades, in storage upgrades, and delivered LRIP of non-future systems assigned to

a product family p in time t in the conventional time horizon.

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

$$PfPurchAndUpgExpenses_{p,t} =$$

$$\sum_{\substack{i,c: \\ i \in PurchasableSys \\ i \in ProductFamily_p}} NumSysPurch_{i,c,t+PurchDelay_i+1} * PrePurchaseCost_i$$

$$+ \sum_{\substack{i,c: \\ i \in PurchasableSys \\ i \in ProductFamily_p}} NumSysPurch_{i,c,t+PurchDelay_i} * PurchCost_i$$

$$+ \sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in MissionUpg \\ j \in ProductFamily_p}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j}$$

$$+ \sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in MissionUpg \\ j \in ProductFamily_p}} NumSysInMissionUpg_{i,j,m,c,t+UpgDelay_{i,j}} * UpgCost_{i,j}$$

$$+ \sum_{\substack{i,j,c: \\ (i,j) \in StorageUpg: \\ j \in ProductFamily_p}} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}+1} * PreUpgCost_{i,j}$$

$$+ \sum_{\substack{i,j,c: \\ (i,j) \in StorageUpg: \\ j \in ProductFamily_p}} iNumInStorageUpg_{i,j,c,t+UpgDelay_{i,j}} * UpgCost_{i,j}$$

$$+ \sum_{\substack{i,c: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i}+1 \leq T}} LripPreCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}+1}$$

$$+ \sum_{\substack{i,c: \\ (p,i) \in LripProfiles \\ t+LripDelay_{p,i} \leq T}} LripCost_{p,i} * iLripDelivered_{c,p,i,t+LripDelay_{p,i}} \tag{5.23}$$

- $NonPurchAndUpgPfExpenses_{p,t}$ denotes the amount spent by product family p on procurement active costs, startup costs, and LRIP produced but not delivered in time

t.

$\forall p, t \leq T$

$$\begin{aligned}
&NonPurchAndUpgPfExpenses_{p,t} = \\
&\quad bPfActive_{p,t} * PfProcureActiveCost_p \\
&\quad + PfStartupCost_{p,t} \\
&\quad + \sum_{\substack{c,i: \\ (p,i)\in LripProfiles \\ t+LripDelay_{p,i}+1\leq T}} iLripProduced_{c,p,i,t+LripDelay_{p,i}+1} * LripPreCost_{p,i} \\
&\quad - \sum_{\substack{c,i: \\ (p,i)\in LripProfiles \\ t+LripDelay_{p,i}+1\leq T}} iLripDelivered_{c,p,i,t+LripDelay_{p,i}+1} * LripPreCost_{p,i} \\
&\quad + \sum_{\substack{c,i: \\ (p,i)\in LripProfiles \\ t+LripDelay_{p,i}\leq T}} iLripProduced_{c,p,i,t+LripDelay_{p,i}} * LripCost_{p,i} \\
&\quad - \sum_{\substack{c,i: \\ (p,i)\in LripProfiles \\ t+LripDelay_{p,i}\leq T}} iLripDelivered_{c,p,i,t+LripDelay_{p,i}} * LripCost_{p,i}
\end{aligned} \tag{5.24}$$

- $AllNonPurchAndUpgPfExpenses_t$ denotes the amount spent by all product families on procurement active costs, startup costs, and LRIP produced but not delivered in time t.

$\forall t \leq T$

$$AllNonPurchAndUpgPfExpenses_t = \sum_p NonPurchAndUpgPfExpense_{p,t} \tag{5.25}$$

- $ProcureExpense_t$ 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.

$\forall t \leq T$

$$ProcureExpense_t = PurchAndUpgExpense_t + AllNonPurchAndUpgPfExpense_t \tag{5.26}$$

- $FutureComponentPurchAndUpgExpense_{c,t}$ denotes the amount spent solely on purchases, in mission upgrades, and in storage upgrades of future systems in addition to purchases and upgrades of coasting non-future systems for component c in time t within the conventional and extended time horizons. Here the binary parameter

$\gamma_{i,\mathcal{J},m}$ equals 1 if $FutureTransitionLongLead_{i,\mathcal{J},m} > 0$ and 0 otherwise.

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

$$
\begin{aligned}
&FutureComponentPurchAndUpgExpense_{c,t} = \\
&\sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ t^*=t+FutureTransitionDelay_{i,\mathcal{J},m}+\gamma_{i,\mathcal{J},m}\leq\mathcal{T}}} \left(\begin{array}{l} NumSysReplaced_{i,\mathcal{J},m,c,t^*} \\ *FutureTransitionCost_{i,\mathcal{J},m,c} \\ *FutureTransitionLongLead_{i,\mathcal{J},m,c} \end{array} \right) \\
&+ \sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ t^*=t+FutureTransitionDelay_{i,\mathcal{J},m}\leq\mathcal{T}}} \left(\begin{array}{l} NumSysReplaced_{i,\mathcal{J},m,c,t^*} \\ *FutureTransitionCost_{i,\mathcal{J},m,c} \\ *(1-FutureTransitionLongLead_{i,\mathcal{J},m,c}) \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastablePurchTransitions \\ t^*=t+PurchDelay_i+1}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ *PrePurchaseCost_i * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastablePurchTransitions \\ t^*=t+PurchDelay_i}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ *PurchaseCost_i * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastableUpgTransitions \\ t^*=t+UpgDelay_i+1}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ *PreUpgCost_{i,j} * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastableUpgTransitions \\ t^*=t+UpgDelay_i}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ *UpgCost_{i,j} * SysPerGp_m \end{array} \right)
\end{aligned} \tag{5.27}
$$

- $FutureComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t}$ denotes the amount spent solely on purchases, in mission upgrades, and in storage upgrades of future systems in addition to purchases and upgrades of coasting non-future systems for component c for all systems in product families without earmarks in time t within the conventional and extended time horizons. Here the binary parameter $\gamma_{i,\mathcal{J},m}$ equals 1 if

$FutureTransitionLongLead_{i,\mathcal{J},m} > 0$ and 0 otherwise.

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

$$
\begin{aligned}
&FutureComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t} = \\
&\sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ t^*=t+FutureTransitionDelay_{i,\mathcal{J},m}+\gamma_{i,\mathcal{J},m}\leq\mathcal{T}}} \left(\begin{array}{l} NumSysReplaced_{i,\mathcal{J},m,c,t^*} \\ *FutureTransitionCost_{i,\mathcal{J},m,c} \\ *FutureTransitionLongLead_{i,\mathcal{J},m,c} \end{array} \right) \\
&+ \sum_{\substack{i,\mathcal{J},m: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ t^*=t+FutureTransitionDelay_{i,\mathcal{J},m}\leq\mathcal{T}}} \left(\begin{array}{l} NumSysReplaced_{i,\mathcal{J},m,c,t^*} \\ *FutureTransitionCost_{i,\mathcal{J},m,c} \\ *(1-FutureTransitionLongLead_{i,\mathcal{J},m,c}) \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastablePurchTransitions \\ t^*=t+PurchDelay_i+1 \\ j\notin SysInPfWithEarmark_t}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ *PrePurchaseCost_i * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastablePurchTransitions \\ t^*=t+PurchDelay_i \\ j\notin SysInPfWithEarmark_t}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ *PurchaseCost_i * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastableUpgTransitions \\ t^*=t+UpgDelay_i+1 \\ j\notin SysInPfWithEarmark_t}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ *PreUpgCost_{i,j} * SysPerGp_m \end{array} \right) \\
&+ \sum_{\substack{i,j,m: \\ (i,j,m,c)\in CoastableUpgTransitions \\ t^*=t+UpgDelay_i \\ j\notin SysInPfWithEarmark_t}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ *UpgCost_{i,j} * SysPerGp_m \end{array} \right)
\end{aligned} \tag{5.28}
$$

- $FuturePurchAndUpgExpense_t$ denotes the amount spent solely on purchases, in mission upgrades, and in storage upgrades of future systems and purchases and upgrades of coasting non-future systems for all components in time t within the conventional and extended time horizons.

$\forall t \leq T$

$$FuturePurchAndUpgExpense_t = \sum_c FutureComponentPurchAndUpgExpense_{c,t} \tag{5.29}$$

- $FutureAllNonPurchAndUpgProgramExpenses_t$ denotes the amount spent by future programs on procurement active costs, startup costs, and LRIP in time t. Here, the

binary parameter $\eta_{\mathcal{J}}$ equals 1 if $FutureSysLripLongLead_{\mathcal{J}} > 0$ and 0 otherwise.

$$\begin{aligned}
&\forall t \leq \mathcal{T} \\
&\quad FutureAllNonPurchAndUpgProgramExpenses_t = \\
&\qquad \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureProgramStartupCostSchedule_{\mathcal{F},t} \\
&\qquad + \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureProgramActiveCostSchedule_{\mathcal{F},t} \\
&\qquad + \sum_{\substack{\mathcal{J},c: \\ t^*=t+FutureSysLripDelay_{\mathcal{J}}+\eta_{\mathcal{J}} \leq \mathcal{T}}} \left(\begin{array}{l} bFutureSysComponent_{\mathcal{J},c} \\ *FutureSysLripCost_{\mathcal{J}} \\ *FutureSysLripProfile_{\mathcal{F},c,t^*} \\ *FutureSysLripLongLead_{\mathcal{J}} \end{array} \right) \\
&\qquad + \sum_{\substack{\mathcal{J},c: \\ t^*=t+FutureSysLripDelay_{\mathcal{J}} \leq \mathcal{T}}} \left(\begin{array}{l} bFutureSysComponent_{\mathcal{J},c} \\ *FutureSysLripCost_{\mathcal{J}} \\ *FutureSysLripProfile_{\mathcal{F},c,t^*} \\ *(1 - FutureSysLripLongLead_{\mathcal{J}}) \end{array} \right)
\end{aligned} \tag{5.30}$$

- $FutureNonPurchAndUpgPfExpenses_{p,t}$ denotes the amount spent by product family p in time t for procurement active costs associated with coasting systems.

$$\begin{aligned}
&\forall p, t \leq \mathcal{T} \\
&\quad FutureNonPurchAndUpgPfExpenses_{p,t} = \\
&\qquad \begin{cases} 0 & t \leq T \\ bPfActive_{p,t} * PfProcureActiveCost_p & t > T \end{cases}
\end{aligned} \tag{5.31}$$

- $FutureAllNonPurchAndUpgPfExpenses_t$ denotes the amount spent by all product families in time t for procurement active costs associated with coasting systems.

$$\begin{aligned}
&\forall t \leq \mathcal{T} \\
&\quad FutureAllNonPurchAndUpgPfExpenses_t = \\
&\qquad \sum_{p} FutureNonPurchAndUpgPfExpense_t
\end{aligned} \tag{5.32}$$

- $FuturePfPurchAndUpgExpenses_{p,t}$ denotes the amount spent by product family p

on the procurement of coasting systems in time t.

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

$$FuturePfPurchAndUpgExpenses_{p,t} =$$
$$\sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in CoastablePurchTransitions \\ t^* = t + PurchDelay_i + 1 \\ j \in ProductFamily_p}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ * PrePurchaseCost_i * SysPerGp_m \end{array} \right)$$
$$+ \sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in CoastablePurchTransitions \\ t^* = t + PurchDelay_i \\ j \in ProductFamily_p}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ * PurchaseCost_i * SysPerGp_m \end{array} \right)$$
$$+ \sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in CoastableUpgTransitions \\ t^* = t + UpgDelay_i + 1 \\ j \in ProductFamily_p}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ * PreUpgCost_{i,j} * SysPerGp_m \end{array} \right)$$
$$+ \sum_{\substack{i,j,m,c: \\ (i,j,m,c) \in CoastableUpgTransitions \\ t^* = t + UpgDelay_i \\ j \in ProductFamily_p}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ * UpgCost_{i,j} * SysPerGp_m \end{array} \right) \quad (5.33)$$

- $FutureProcureExpense_t$ denotes the amount spent on procurement of future systems (in storage upgrades, in mission upgrades, purchases, future programs, and LRIP) and coasting systems for all components at time t within the conventional and extended time horizons.

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

$$FutureProcureExpense_t =$$
$$FuturePurchAndUpgExpense_t$$
$$+ FutureAllNonPurchAndUpgPfExpenses_t$$
$$+ FutureAllNonPurchAndUpgProgramExpenses_t \quad (5.34)$$

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

$\forall t \leq T$

$$OSExpense_t = \sum_{(i,m,c) \in Roles} SysOSCost_{i,m} * NumSysInMission_{i,m,c,t} \quad (5.35)$$

- $FutureOSExpense_t$ 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.

$$
\begin{aligned}
&\forall t \leq \mathcal{T} \\
&\quad FutureOSExpense_t = \\
&\qquad \sum_{\substack{i,\mathcal{J},m,c,t^*: \\ (i,\mathcal{J},m)\in FutureTransitions \\ t^*\leq t}} (NumSysReplaced_{i,\mathcal{J},m,c,t^*} * FutureSysOSCost_{\mathcal{J}}) \\
&\qquad + \begin{cases} 0 & t \leq T \\ \\ OSExpense_T & T < t \leq \mathcal{T} \end{cases} \\
&\qquad - \sum_{\substack{i,\mathcal{J},m,c,t^*: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ (i,m,c)\in Roles \\ T<t^*\leq t}} (NumSysReplaced_{i,\mathcal{J},m,c,t^*} * SysOSCost_{i,m}) \\
&\qquad + \sum_{\substack{j,i,m,c,t^* \\ (j,i,m,c)\in CoastablePurchTransitions \\ (i,m,c)\in Roles \\ t^*\leq t}} \left(\begin{array}{l} iNumGpCoastingPurch_{j,i,m,c,t^*} \\ *SysPerGp_m * SysOSCost_{i,m} \end{array} \right) \\
&\qquad + \sum_{\substack{j,i,m,c,t^* \\ (j,i,m,c)\in CoastableUpgTransitions \\ (i,m,c)\in Roles \\ t^*\leq t}} \left(\begin{array}{l} iNumGpCoastingUpg_{j,i,m,c,t^*} \\ *SysPerGp_m * SysOSCost_{i,m} \end{array} \right) \\
&\qquad - \sum_{\substack{i,j,m,c,t^* \\ (i,j,m,c)\in CoastablePurchTransitions \\ (i,m,c)\in Roles \\ t^*\leq t}} \left(\begin{array}{l} iNumGpCoastingPurch_{i,j,m,c,t^*} \\ *SysPerGp_m * SysOSCost_{i,m} \end{array} \right) \\
&\qquad - \sum_{\substack{i,j,m,c,t^* \\ (i,j,m,c)\in CoastableUpgTransitions \\ (i,m,c)\in Roles \\ t^*\leq t}} \left(\begin{array}{l} iNumGpCoastingUpg_{i,j,m,c,t^*} \\ *SysPerGp_m * SysOSCost_{i,m} \end{array} \right)
\end{aligned} \tag{5.36}
$$

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

$$
\begin{aligned}
&\forall p \in PfWithRdte, t \leq T \\
&\quad RdteEffortExpense_{p,t} = \\
&\qquad bPfActive_{p,t} * PfRdteActiveCost_p \\
&\qquad + \sum_{d\in AllowedRdteDelays_p} bRdteDelay_{p,d} * RdteCostProfiles_{p,d,t}
\end{aligned} \tag{5.37}
$$

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

within the conventional time horizon.

$\forall t \leq T$

$$RdteExpense_t = \sum_{p \in PfWithRdte} RdteEffortExpense_{p,t} \tag{5.38}$$

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

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

$$FutureRdteExpense_t = \sum_{\mathcal{F}} bFutureProgram_{\mathcal{F}} * FutureProgramRdteCostSchedule_{\mathcal{F},t} + \begin{cases} 0 & t \leq T \\ \sum_{p \in PfWithRdteActive} bPfActive_{p,t} * PfRdteActiveCost_p & T < t \leq \mathcal{T} \end{cases} \tag{5.39}$$

- $CombinedExpense_t$ denotes the combined amount spent on any set of the procurement, O&S, and RDT&E expenses 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.

$\forall t \leq T$

$$CombinedExpense_t = b_{Proc} * ProcureExpense_t + b_{OS} * OSExpense_t + b_{Rdte} * RdteExpense_t \tag{5.40}$$

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

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

$$FutureCombinedExpense_t = b_{Proc} * FutureProcureExpense_t + b_{OS} * FutureOSExpense_t + b_{Rdte} * FutureRdteExpense_t \tag{5.41}$$

- $CombinedComponentEarmarkSpent_t$ denotes the combined amount of component earmarks spent at time t.

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

$$CombinedComponentEarmarkSpent_t = \sum_c fComponentEarmarkSpent_{c,t} \tag{5.42}$$

- $CombinedPfEarmarkSpentProduction_t$ denotes the combined amount of product family earmarks spent at time t on all production associated costs.

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

$$CombinedPfEarmarkSpentProduction_t = \sum_p fPfEarmarkSpentProduction_{p,t} \tag{5.43}$$

- $CombinedPfEarmarkSpentRdte_t$ denotes the combined amount of product family earmarks spent at time t on RDT&E.

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

$$CombinedPfEarmarkSpentRdte_t = \sum_p fPfEarmarkSpentRdte_{p,t} \tag{5.44}$$

Production

- $NumSysInProduction_{i,t}$ 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 including coasting systems in all components.

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

$$\begin{aligned}
&NumSysInProduction_{i,t} = \\
&\quad \begin{cases} \sum\limits_{\substack{c \\ PurchProdDelay_i=0}} NumSysPurch_{i,c,t} & t \leq T \\ 0 & t > T \end{cases} \\
&\quad + \sum_{\substack{c,t^*: \\ t<t^* \leq t+PurchProdDelay_i \\ t^* \leq T \\ PurchProdDelay_i>0}} NumSysPurch_{i,c,t^*} \\
&\quad + \begin{cases} \sum\limits_{\substack{j,m,c: \\ (j,i,m,c) \in MissionUpg \\ UpgProdDelay_{j,i}=0}} NumSysInMissionUpg_{j,i,m,c,t} & t \leq T \\ 0 & t > T \end{cases} \\
&\quad + \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c) \in MissionUpg \\ t<t^* \leq t+UpgProdDelay_{j,i} \\ t^* \leq T \\ UpgProdDelay_{j,i}>0}} NumSysInMissionUpg_{j,i,m,c,t^*}
\end{aligned}$$

$$
\begin{aligned}
&+ \begin{cases} \displaystyle\sum_{\substack{j,c: \\ (j,i)\in StorageUpg \\ UpgProdDelay_{j,i}=0}} iNumInStorageUpg_{j,i,c,t} & t \le T \\ 0 & t > T \end{cases} \\
&+ \sum_{\substack{j,c,t^*: \\ (j,i)\in StorageUpg \\ t<t^*\le t+UpgProdDelay_{j,i} \\ t^*\le T \\ UpgProdDelay_{j,i}>0}} iNumInStorageUpg_{j,i,c,t^*} \\
&+ \begin{cases} 0 & t \le T \\ \displaystyle\sum_{\substack{j,m,c: \\ (j,i,m,c)\in CoastablePurchTransitions \\ PurchProdDelay_i=0}} iNumGpCoastingPurch_{j,i,m,c,t} * SysPerGp_m & t > T \end{cases} \\
&+ \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c)\in CoastablePurchTransitions \\ t<t^*\le t+PurchProdDelay_i \\ t^*>T \\ PurchProdDelay_i>0}} iNumGpCoastingPurch_{j,i,m,c,t^*} * SysPerGp_m \\
&+ \begin{cases} 0 & t \le T \\ \displaystyle\sum_{\substack{j,m,c: \\ (j,i,m,c)\in CoastableUpgTransitions \\ UpgProdDelay_{j,i}=0}} iNumGpCoastingUpg_{j,i,m,c,t} * SysPerGp_m & t > T \end{cases} \\
&+ \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c)\in CoastableUpgTransitions \\ t<t^*\le t+UpgProdDelay_{j,i} \\ t^*>T \\ UpgProdDelay_{j,i}>0}} iNumGpCoastingUpg_{j,i,m,c,t^*} * SysPerGp_m
\end{aligned} \tag{5.45}
$$

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

upgrades, and purchases.

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

$$\begin{aligned}
NumSysInAdminPeriod_{i,t} = & \\
& \sum_{\substack{c,t^* \\ t+PurchProdDelay_i < t^* \leq t+PurchDelay_i \\ t^* \leq T}} NumSysPurch_{i,c,t^*} \\
& + \sum_{\substack{j,c,t^*: \\ (j,i) \in StorageUpg \\ t+UpgProdDelay_{j,i} < t^* \leq t+UpgDelay_{j,i} \\ t^* \leq T}} iNumInStorageUpg_{j,i,c,t^*} \\
& + \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c) \in MissionUpg \\ t+UpgProdDelay_{j,i} < t^* \leq t+UpgDelay_{j,i} \\ t^* \leq T}} NumSysInMissionUpg_{j,i,m,c,t^*} \\
& + \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c) \in CoastablePurchTransitions \\ t+PurchProdDelay_i < t^* \leq t+PurchDelay_i \\ t^* > T}} iNumGpCoastingPurch_{j,i,m,c,t^*} * SysPerGp_m \\
& + \sum_{\substack{j,m,c,t^*: \\ (j,i,m,c) \in CoastableUpgTransitions \\ t+UpgProdDelay_{j,i} < t^* \leq t+UpgDelay_{j,i} \\ t^* > T}} iNumGpCoastingUpg_{j,i,m,c,t^*} * SysPerGp_m
\end{aligned} \tag{5.46}$$

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

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

$$\begin{aligned}
NumSysDeliveredComponent_{i,c,t} = & \\
& \sum_{\substack{i^* \in PurchasableSys: \\ i^* = i}} NumSysPurch_{i^*,c,t} \\
& + \sum_{\substack{j: \\ (j,i) \in StorageUpg}} iNumInStorageUpg_{j,i,c,t} \\
& + \sum_{\substack{j,m: \\ (j,i,m,c) \in MissionUpg}} NumSysInMissionUpg_{j,i,m,c,t}
\end{aligned} \tag{5.47}$$

- $NumSysDelivered_{i,t}$ 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 in all components.

$$\forall i, t \leq T$$
$$NumSysDelivered_{i,t} = \sum_{c} NumSysDeliveredComponent_{i,c,t} \tag{5.48}$$

- $NumSysDeliveredIncludingLripProduced_{i,t}$ 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, purchases, and LRIP produced (icluding those LRIP systems that are never available for fielding to a mission).

$$\forall i, t \leq T$$
$$NumSysDeliveredIncludingLripProduced_{i,t} = NumSysDelivered_{i,t} + \sum_{\substack{p: \\ (p,i) \in LripProfiles}} LripProduced_{p,i,t} \tag{5.49}$$

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

$$\forall i \in CoastableSys, T < t \leq \mathcal{T}$$
$$NumCoastingSysDelivered_{i,t} = \sum_{\substack{j,m,c: \\ (j,i,m,c) \in CoastablePurchTransitions}} iNumGpCoastingPurch_{j,i,m,c,t} * SysPerGp_m + \sum_{\substack{j,m,c: \\ (j,i,m,c) \in CoastableUpgTransitions}} iNumGpCoastingUpg_{j,i,m,c,t} * SysPerGp_m \tag{5.50}$$

Product Families

- $PfSysFielded_{p,c,t}$ denotes the total number of systems fielded from product family p to component c in conventional time period t.

$$\forall p \in PfWithRatios, c, t \leq T$$
$$PfSysFielded_{p,c,t} = \sum_{\substack{i \in ProductFamily_p \\ j,m:(j,i,m,c) \in Transitions}} NumSysFromStorage_{j,i,m,c,t} + \sum_{\substack{i \in ProductFamily_p \\ j,m:(j,i,m,c) \in MissionUpg}} NumSysInMissionUpg_{j,i,m,c,t} \tag{5.51}$$

- $PfSysFieldedWindow_{p,c,t}$ denotes the total number of systems fielded from product family p to component c over the time periods given by $PfRatioWindow_p$ starting in time period t.

$$\forall p \in PfWithRatios, c, t + PfRatioWindow_p - 1 \leq T$$
$$PfSysFieldedWindow_{p,c,t} = \sum_{\substack{t^*: \\ t \leq t^* \leq t + PfRatioWindow_p - 1}} PfSysFielded_{p,c,t^*} \tag{5.52}$$

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. The "age" phase objective **minimizes** the number of system-year economic useful life violations. Similarly, the "yearly budget" phase objective **minimizes** the amount of yearly budget violations and the "horizon budget" phase objective **minimizes** the amount of horizon budget 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 in each component 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. A detailed description of the optimization phases can be found in the "Mission Priority Tiers" section of Chapter 1.

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," "final mandate," or "product family" deficit variables throughout the planning horizon. This takes on value zero

only if all schedule mandates are met.

$$
\begin{aligned}
TierScheduleDeficits = & \\
& \sum_{\substack{i,m,t: \\ (i,m)\in SysMissions \\ MissionTier_m = CurrentTier \\ t \leq \mathcal{T}}} iModernizedDeficit_{i,m,t} \\
& + \sum_{\substack{i,m,c: \\ (i,m,c)\in Roles \\ MissionTier_m = CurrentTier}} iFinalMandateDeficit_{i,m,c} \\
& + \sum_{\substack{\mathcal{J},c: \\ MissionTier_m = MissionTier_{FutureMissionMap_{\mathcal{J}}}}} iFutureSysMandateDeficit_{\mathcal{J},c} \\
& + \sum_{\substack{p,t: \\ t \leq T}} iPfDeliveryDeficit_{p,t}
\end{aligned}
\tag{5.53}
$$

- $TierAgeOverages$ denotes the sum of the system-year overages variable throughout the planning horizon. This takes on value zero only if all economic useful life constraints are met.

$$
TierAgeOverages = \sum_{\substack{i,m,c,t: \\ (i,m,c)\in Roles \\ MissionTier_m = CurrentTier \\ t \leq \mathcal{T}}} iSysYearAgeOverages_{i,m,c,t} \tag{5.54}
$$

- $YearlyBudgetOverruns$ denotes the sum of all budget overage amounts for all yearly budget types throughout the conventional and extended planning horizons. This takes

on value zero only if all budgets are satisfied.

$$
\begin{aligned}
YearlyBudgetOverruns = \\
&\sum_{t \leq T} fProcureBudgetOverrun_t \\
&+ \sum_{t \leq T} fOSBudgetOverrun_t \\
&+ \sum_{t \leq T} fRdteBudgetOverrun_t \\
&+ \sum_{t \leq T} fCombinedBudgetOverrun_t \\
&+ \sum_{T < t \leq \mathcal{T}} fFutureProcureBudgetOverrun_t \\
&+ \sum_{T < t \leq \mathcal{T}} fFutureOSBudgetOverrun_t \\
&+ \sum_{T < t \leq \mathcal{T}} fFutureRdteBudgetOverrun_t \\
&+ \sum_{T < t \leq \mathcal{T}} fFutureCombinedBudgetOverrun_t
\end{aligned} \tag{5.55}
$$

- $TotalBudgetOverruns$ denotes the sum of all budget overage amounts for all horizon budget types. This takes on value zero only if all budgets are satisfied.

$$
\begin{aligned}
TotalBudgetOverruns = \\
&fTotalProcureBudgetOverrun \\
&+ fTotalOSBudgetOverrun \\
&+ fTotalRdteBudgetOverrun \\
&+ fTotalCombinedBudgetOverrun
\end{aligned} \tag{5.56}
$$

- $TotalPerformance$ denotes the cumulative performance of all systems in the fleet. This includes future systems and coasting systems throughout the conventional and extended time horizons.

$$
\begin{aligned}
TotalPerformance = \\
&\sum_{\substack{i,m,c,t: \\ (i,m,c) \in Roles \\ t \leq \mathcal{T}}} \alpha_{i,m} * NumSysInMission_{i,m,c,t} \\
&+ \sum_{\substack{i,\mathcal{J},m,c,t: \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ t \leq \mathcal{T}}} \left(\begin{array}{l} \alpha_{i,\mathcal{J},m,c} * NumSysReplaced_{i,\mathcal{J},m,c,t} \\ *(\mathcal{T} - t + 1) \end{array} \right)
\end{aligned} \tag{5.57}
$$

- $CostPhaseExpenses$ 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.

$$\begin{aligned} CostPhaseExpenses = \\ \sum_{t \leq T} \left(\begin{array}{l} b_{CP-Proc} * ProcureExpense_t + b_{CP-OS} * OSExpense_t \\ + b_{CP-Rdte} * RdteExpense_t \end{array} \right) \\ + \sum_{t \leq \mathcal{T}} \left(\begin{array}{l} b_{CP-Proc} * FutureProcureExpense_t + b_{CP-OS} * FutureOSExpense_t \\ + b_{CP-Rdte} * FutureRdteExpense_t \end{array} \right) \end{aligned} \tag{5.58}$$

- $ObjFunVal$ denotes the objective function that either minimizes schedule violations, minimizes the system-year age overages, minimizes yearly budget overruns, minimizes horizon budget overruns, maximizes cumulative fleet performance, or minimizes the cumulative combined fleet cost, depending on the current phase. The $SchedulePhase$, $AgePhase$, $YearlyBudgetPhase$, $HorizonBudgetPhase$, $PerformancePhase$, 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.

$$\begin{aligned} Maximize\ ObjFunVal = \\ & - SchedulePhase * TierScheduleDeficits \\ & - AgePhase * TierAgeOverages \\ & - YearlyBudgetPhase * YearlyBudgetOverruns \\ & - HorizonBudgetPhase * TotalBudgetOverruns \\ & + PerformancePhase * TotalPerformance \\ & - CostPhase * CostPhaseExpenses \end{aligned} \tag{5.59}$$

This page intentionally left blank.

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.

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

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

$$TierAgeOverages \leq AgeOverrunBound \tag{6.2}$$

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

$$YearlyBudgetOverruns \leq YearlyBudgetOverrunBound \tag{6.3}$$

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

$$TotalBudgetOverruns \leq HorizonBudgetOverrunBound \tag{6.4}$$

- 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.

$$TotalPerformance >= MinimumPerformance - 0.001 \tag{6.5}$$

- For phases after the cost phase, ensure that the combined fleet cost for all costs included in the cost phase does not increase from the value found in the cost phase. Prior to and during the cost phase, the $MaximumCost$ parameter is not restrictive. This partially addresses the **Tier Phases** business rule.

$$CostPhaseExpenses <= MaximumCost \tag{6.6}$$

- If hard limits are placed on any of the phase objective metrics, the corresponding constraint is enforced during *all* phases. If no limit is defined, then the constraint is not enforced. This addresses the **Hard Limits on Tier Phases** business rule.

$$TierScheduleDeficits \leq ScheduleViolationLimit \tag{6.7}$$
$$TierAgeOverages \leq AgeViolationLimit \tag{6.8}$$
$$YearlyBudgetOverruns \leq YearlyBudgetOverrunLimit \tag{6.9}$$
$$TotalBudgetOverruns \leq TotalBudgetOverrunLimit \tag{6.10}$$
$$TotalPerformance \geq PerformanceLimit \tag{6.11}$$
$$CostPhaseExpenses \leq CostLimit \tag{6.12}$$

- Constraints (6.13)–(6.21) ensure that no modernization occurs for missions in lower-priority tiers than the current tier and address the **Priority Tiers** business rule.

$$\forall (i,j,m,c) \in MissionUpg, t \leq T \text{ where } MissionTier_m > CurrentTier$$
$$NumSysInMissionUpg_{i,j,m,c,t} = 0 \tag{6.13}$$

$$\forall (i,j,m,c) \in Transitions, t \leq T \text{ where } MissionTier_m > CurrentTier$$
$$NumSysFromStorage_{i,j,m,c,t} = 0 \tag{6.14}$$

$$\forall (i,m,c) \in CoastableRoles \text{ where } MissionTier_m > CurrentTier$$
$$iCoastingLevel(i,m,c) = 0 \tag{6.15}$$

$$\forall (i,j,m,c) \in CoastablePurchTransitions,$$
$$T < t \leq \mathcal{T} \text{ where } MissionTier_m > CurrentTier$$
$$iNumGpCoastingPurch_{i,j,m,c,t} = 0 \tag{6.16}$$

$$\forall (i,j,m,c) \in CoastableUpgTransitions,$$
$$T < t \leq \mathcal{T} \text{ where } MissionTier_m > CurrentTier$$
$$iNumGpCoastingUpg_{i,j,m,c,t} = 0 \tag{6.17}$$

$$\forall (i,\mathcal{J},m,c) \in FutureTransitions, t \leq \mathcal{T} \text{ where } MissionTier_m > CurrentTier$$
$$iNumGpReplaced_{i,\mathcal{J},m,c,t} = 0 \tag{6.18}$$

$\forall(i,m) \in SysMissions, t \leq \mathcal{T}$ where $MissionTier_m > CurrentTier$

$$iModernizedDeficit_{i,m,t} = 0 \tag{6.19}$$

$\forall(i,m,c) \in MandatedRoles$ where $MissionTier_m > CurrentTier$

$$iFinalMandateDeficit_{i,m,c} = 0 \tag{6.20}$$

$\forall \mathcal{J}, c$ where $MissionTier_{FutureMissionMap_{\mathcal{J}}} > CurrentTier$

$$iFutureSysMandateDeficit_{\mathcal{J},c} = 0 \tag{6.21}$$

- Constraints (6.22)–(6.30) 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.

$\forall(i,j,m,c,t,N) \in fixedSysInMissionUpg$

$$iNumGpInMissionUpg_{i,j,m,c,t} = N \tag{6.22}$$

$\forall(i,j,m,c,t,N) \in fixedSysFromStorage$

$$iNumGpFromStorage_{i,j,m,c,t} = N \tag{6.23}$$

$\forall(i,m,c,N) \in fixedCoastingLevel$

$$iCoastingLevel(i,m,c) = N \tag{6.24}$$

$\forall(i,j,m,c,t,N) \in fixedGpCoastingPurch$

$$iNumGpCoastingPurch_{i,j,m,c,t} = N \tag{6.25}$$

$\forall(i,j,m,c,t,N) \in fixedGpCoastingUpg$

$$iNumGpCoastingUpg_{i,j,m,c,t} = N \tag{6.26}$$

$\forall(i,\mathcal{J},m,c,t,N) \in fixedGpReplaced$

$$iNumGpReplaced_{i,\mathcal{J},m,c,t} = N \tag{6.27}$$

$\forall(i,m,t,N) \in fixedModernizedDeficit$

$$iModernizedDeficit_{i,m,t} = N \tag{6.28}$$

$\forall(i,m,c,N) \in fixedFinalMandateDeficit$

$$iFinalMandateDeficit_{i,m,c} = N \tag{6.29}$$

$\forall(\mathcal{J},c,N) \in fixedFinalFutureMandateDeficit$

$$iFutureSysMandateDeficit_{\mathcal{J},c} = N \tag{6.30}$$

- Constraints (6.31)–(6.33) ensure that if one mission in a specific component precedes another another mission in a specific component, 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 **Component Mission Succession** business rule.

$$\begin{aligned}
&\forall (c,m) \in SuccessorComponentMissions, t \leq T \\
&\quad \sum_{i,j:\ (i,j,m,c) \in Transitions} iNumGpFromStorage_{i,j,m,c,t} \\
&\qquad + \sum_{i,j:\ (i,j,m,c) \in MissionUpg} iNumGpInMissionUpg_{i,j,m,c,t} \\
&\qquad \leq maxGpTransitionLimit * bMissionCanField_{m,c,t}
\end{aligned} \tag{6.31}$$

$$\begin{aligned}
&\forall (c,m) \in SuccessorComponentMissions, t \leq T \\
&\quad 1 - bMissionCanField_{m,c,t} \\
&\qquad \leq \sum_{\substack{i,m^*,c^*: \\ (i,m^*,c^*) \in Roles \\ (c^*,m^*,c,m) \in ComponentMissionSuccessions \\ InitialGpInMission_{i,m^*,c^*} > 0}} NumGpInMission_{i,m^*,c^*,t} \\
&\qquad + \sum_{\substack{i,j,m^*,c^*,t^*: \\ (i,j,m^*,c^*) \in Transitions \\ (c^*,m^*,c,m) \in ComponentMissionSuccessions \\ t \leq t^* \leq T}} iNumGpFromStorage_{i,j,m^*,c^*,t^*} \\
&\qquad + \sum_{\substack{i,j,m^*,c^*,t^*: \\ (i,j,m^*,c^*) \in MissionUpg \\ (c^*,m^*,c,m) \in ComponentMissionSuccessions \\ t \leq t^* \leq T}} iNumGpInMissionUpg_{i,j,m^*,c^*,t^*}
\end{aligned} \tag{6.32}$$

$$\forall (c,m) \in SuccessorComponentMissions, t \leq T$$

$$\sum_{\substack{m^*,c^*:(c^*,m^*,c,m)\in \\ ComponentMissionSuccessions}} (GpPerComponentMission_{c^*,m^*} * (maxPathLength_{m^*}+1))$$

$$* (1 - bMissionCanField_{m,c,t})$$

$$\geq \sum_{\substack{i,m^*,c^*: \\ (i,m^*,c^*)\in Roles \\ (c^*,m^*,c,m)\in ComponentMissionSuccessions \\ InitialGpInMission_{i,m^*,c^*}>0}} NumGpInMission_{i,m^*,c^*,t}$$

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

$$+ \sum_{\substack{i,j,m^*,c^*,t^*: \\ (i,j,m^*,c^*)\in MissionUpg \\ (c,^*,m^*,c,m)\in ComponentMissionSuccessions \\ t\leq t^*\leq T}} iNumGpInMissionUpgrade_{i,j,m^*,c^*,t^*}$$

(6.33)

System Flow Conservation Constraints

- This constraint implies that the number of systems i in component c 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.

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

$$NumSysInStorageExchangeable_{i,c,t} \geq 0 \tag{6.34}$$

- This constraint ensures that the number of systems i in mission m in component c 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. This constraint also fulfills the **1-Year Duty Minimum** business rule.

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

$$NumSysInMissionExchangeable_{i,m,c,t} \geq 0 \tag{6.35}$$

- This constraint assigns the initial populations of systems in storage to a specific component. Once assigned, these systems can only be fielded to the specified component. This helps fulfill the storage part of the **Initial Populations** business rule.

$$\forall i,c$$

$$\sum_c iInitialInStorage_{c,i} = InitialInStorage_i \tag{6.36}$$

- This constraint limits the pool of potential systems i in component c 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.

$$\begin{aligned}
&\forall i, c, t \leq T \\
&\sum_{\substack{j,t^*: \\ (i,j)\in StorageUpg \\ t^*\leq t+UpgProdDelay_{i,j}}} iNumInStorageUpg_{i,j,c,t^*} \\
&\quad \leq iInitialSysInStorage_{i,c} \\
&\quad + \sum_{\substack{j,m,t^*: \\ (i,j,m,c)\in Transitions \\ t^*\leq t}} NumSysFromStorage_{i,j,m,c,t^*} \\
&\quad + \sum_{\substack{j,t^*: \\ (j,i)\in StorageUpg \\ t^*\leq t}} iNumInStorageUpg_{j,i,c,t^*} \\
&\quad + \sum_{\substack{\mathcal{J},m,t^*: \\ (i,\mathcal{J},m,c)\in FutureTransitionPurch \\ t^*\leq t}} NumSysReplaced_{i,\mathcal{J},m,c,t^*} \qquad (6.37)
\end{aligned}$$

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

$$\begin{aligned}
&\forall i, c, t \leq T \\
&NumSysPurch_{i,c,t} \\
&\quad + \sum_{\substack{j: \\ (j,i)\in StorageUpg}} iNumInStorageUpg_{j,i,c,t} \\
&\quad \leq \sum_{\substack{j,m: \\ (j,i,m,c)\in Transitions}} NumSysFromStorage_{j,i,m,c,t} \qquad (6.38)
\end{aligned}$$

- Constraints (6.39) and (6.40) ensure that any group of systems i that are retired from some mission m in component c in time period t cannot be immediately re-fielded back to the same mission and same component in the same time period. This fulfills the

No Retire and Re-Field business rule.

$$\forall (i,m,c) \in InterimRoles, t \leq T$$
$$GpTransitionLimitPerTime_m * bInterimGpCanField_{i,m,c,t} \geq \sum_{j \in Inflow_{i,m,c}} iNumGpFromStorage_{j,i,m,c,t} \quad (6.39)$$

$$\forall (i,m,c) \in InterimRoles, t \leq T$$
$$GpTransitionLimitPerTime_m * (1 - bInterimGpCanField_{i,m,c,t}) \geq \sum_{j \in Outflow_{i,m,c}} iNumGpFromStorage_{i,j,m,c,t} \quad (6.40)$$

- Constraints (6.41) amd (6.42) ensure that newly purchased or in storage upgraded systems i (excluding "hull" systems) may only remain in storage for $maxTimeAllowedInStorage$ time periods. That is, if a system is delivered to storage by a purchase or an in storage upgrade in time t, then that system must be fielded to a mission in time period $t, t+1, t+2, \ldots,$ or $t + maxTimeAllowedInStorage$. A slack of $\max\{PurchBatchSize_i, maxMissionRequirement_i\} - 1$ is allowed to remain in storage indefinitely and not be fielded. Constraint (6.41) deals with systems that do not have LRIP delivery or initial systems in storage. For systems that do have either LRIP delivered to storage or initial storage inventory, constraint (6.42) ensures that these systems may only remain in storage for $maxTimeAllowedInStorage$ time periods, beginning with the first time period that non-LRIP production is completed. This fulfills the **Fielding New Systems From Storage** business rule.

$$\forall i, c, t \leq T - maxTimeAllowedInStorage \text{ where } i \notin FreeInterimUpgSys \text{ and } i \notin SysWithLripOrInitialStorage$$
$$\sum_{\substack{i^*,t^*: \\ i^* \in PurchasableSys, i^* = i \\ t^* \leq t}} NumSysPurch_{i^*,c,t^*} + \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpg \\ t^* \leq t}} iNumInStrorageUpgraded_{j,i,c,t^*} \leq \sum_{\substack{j,m,t^*: \\ (j,i,m,c) \in Transitions \\ t^* \leq t + maxTimeAllowedInStorage}} NumSysFromStorage_{j,i,m,c,t^*} + \max\{PurchBatchSize_i, maxMissionRequirement_i\} - 1 \quad (6.41)$$

$$\forall i, c, t \leq T - maxTimeAllowedInStorage \text{ where } i \notin FreeInterimUpgSys$$
$$\text{and } i \in SysWithLripOrInitialStorage$$
$$\sum_{\substack{i^*,t^*: \\ i^* \in PurchasableSys, i^*=i \\ t^* \leq t}} NumSysPurch_{i^*,c,t^*}$$
$$+ \sum_{\substack{j,t^*: \\ (j,i) \in StorageUpg \\ t^* \leq t}} iNumInStrorageUpgraded_{j,i,c,t^*}$$
$$+ \sum_{\substack{p,t^*: \\ (p,i) \in LripProfiles \\ t^* < t}} iLripDelivered_{c,p,i,t}$$
$$+ iInitialInStorage_{i,c}$$
$$- TotalSysPopulation * (1 - bSysHasDeliveredComponent_{i,c,t})$$
$$\leq \sum_{\substack{j,m,t^*: \\ (j,i,m,c) \in Transitions \\ t^* \leq t + maxTimeAllowedInStorage}} NumSysFromStorage_{j,i,m,c,t^*}$$
$$+ \max\{PurchBatchSize_i, maxMissionRequirement_i\} - 1 \tag{6.42}$$

- Constraint (6.43) ensures that any system i retired from any mission in component c in time t and placed in storage must remain in storage for time period t. This prevents those systems retired from being re-fielded in the same time period t. This rule does not prevent systems retired from a mission and immediately upgraded to "hull" systems in storage from being upgraded to a different system and being re-fielded in that same time period. This fulfills the **Disallow Instantaneous Cross-Mission Transfers** business rule.

$$\forall i, c, t \leq T$$
$$NumSysInStorage_{i,c,t}$$
$$\geq \sum_{\substack{j,m: \\ (i,j,m,c) \in Transitions}} NumSysFromStorage_{i,j,m,c,t}$$
$$+ \sum_{\substack{\mathcal{J},m,: \\ (i,\mathcal{J},m,c) \in FuturePurchaseTransitions}} NumSysReplaced_{i,\mathcal{J},m,c,t}$$
$$- \sum_{\substack{j: \\ (i,j) \in StorageUpg \\ j \in FreeInterimUpgSys}} iNumInStorageUpg_{i,j,c,t} \tag{6.43}$$

General Scheduling Constraints

- Constraints (6.44)–(6.46) ensure that at the beginning of the planning horizon, no systems are purchased, in mission upgraded, or in storage upgraded for any component 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.

$$\forall i \in PurchasableSys, c, t \leq \begin{cases} PurchDelay_i + 1 & \text{if } PrePurchCost_i > 0 \\ PurchDelay_i & \text{if } PrePurchCost_i = 0 \end{cases}$$
$$iNumBatchesPurch_{i,c,t} = 0 \tag{6.44}$$

$$\forall (i,j,m,c) \in MissionUpg, t \leq \begin{cases} UpgDelay_{i,j} + 1 & \text{if } PreUpgCost_{i,j} > 0 \\ UpgDelay_{i,j} & \text{if } PreUpgCost_{i,j} = 0 \end{cases}$$
$$iNumGpInMissionUpg_{i,j,m,c,t} = 0 \tag{6.45}$$

$$\forall (i,j) \in StorageUpg, c, t \leq \begin{cases} UpgDelay_{i,j} + 1 & \text{if } PreUpgCost_{i,j} > 0 \\ UpgDelay_{i,j} & \text{if } PreUpgCost_{i,j} = 0 \end{cases}$$
$$iNumInStorageUpg_{i,j,c,t} = 0 \tag{6.46}$$

- This constraint ensures that the required percentage of initial systems i in mission m for all components 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.

$$\forall (i,m) \in SysMissions, t \leq \mathcal{T}$$
$$\text{where } ModernPercent_{i,m,t} > 0 \text{ and } MissionTier_m = CurrentTier$$
$$\sum_c GpRetiredFromRoleSoFar_{i,m,c,t} * SysPerGp_m + iModernizedDeficit_{i,m,t} \geq ModernPercent_{i,m,t} * \sum_c InitialSysInMission_{i,m,c} \tag{6.47}$$

- These constraints ensure that the number of groups transitioned for mission m in all components at time t is below a specified limit, fulfilling the **Per-Period Mission**

Modernization Limit business rule.

$$\begin{aligned}
&\forall m, t \leq T \text{ where } GpTransitionLimitPerTime_m < \infty \\
&\qquad \text{and } MissionTier_m = CurrentTier \\
&\qquad \sum_{\substack{i,j,c: \\ (i,j,m,c) \in Transitions}} NumGpTransit_{i,j,m,c,t} \\
&\quad + \sum_{\substack{i,\mathcal{J},c: \\ (i,\mathcal{J},m,c) \in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \\
&\quad \leq GpTransitionLimitPerTime_m
\end{aligned} \tag{6.48}$$

$$\begin{aligned}
&\forall m, T < t \leq \mathcal{T} \text{ where } GpTransitionLimitPerTime_m < \infty \\
&\qquad \text{and } MissionTier_m = CurrentTier \\
&\qquad \sum_{\substack{i,j,c: \\ (i,j,m,c) \in CoastablePurchTransitions}} iNumGpCoastingPurch_{i,j,m,c,t} \\
&\quad + \sum_{\substack{i,j,c: \\ (i,j,m,c) \in CoastableUpgTransitions}} iNumGpCoastingUpg_{i,j,m,c,t} \\
&\quad + \sum_{\substack{i,\mathcal{J},c: \\ (i,\mathcal{J},m,c) \in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \\
&\quad \leq GpTransitionLimitPerTime_m
\end{aligned} \tag{6.49}$$

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

Modernization Limit business rule.

$\forall m$ where $GpTransitionLimitTotal_m < \infty$ and $MissionTier_m = CurrentTier$

$$\sum_{\substack{i,j,c,t: \\ (i,j,m,c) \in Transitions \\ InitialSysInMission_{i,m,c} > 0 \\ t \leq T}} NumGpTransit_{i,j,m,c,t}$$

$$+ \sum_{\substack{i,j,c,t: \\ (i,j,m,c) \in CoastablePurchTransitions \\ InitialSysInMission_{i,m,c} > 0 \\ T < t \leq \mathcal{T}}} iNumGpCoastingPurch_{i,j,c,m,t}$$

$$+ \sum_{\substack{i,j,c,t: \\ (i,j,m,c) \in CoastableUpgTransitions \\ InitialSysInMission_{i,m,c} > 0 \\ T < t \leq \mathcal{T}}} iNumGpCoastingUpg_{i,j,c,m,t}$$

$$+ \sum_{\substack{i,\mathcal{J},c,t: \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ InitialSysInMission_{i,m,c} > 0 \\ t \leq \mathcal{T}}} iNumGpReplaced_{i,\mathcal{J},c,m,t}$$

$$\leq GpTransitionLimitTotal_m \tag{6.50}$$

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

$\forall (i, m, c) \in MandatedRoles$ where $MissionTier_m = CurrentTier$

$$NumSysInMission_{i,m,c,\mathcal{T}} + iFinalMandateDeficit_{i,m,c} * SysPerGp_m \geq FinalMandate_{i,m,c} \tag{6.51}$$

- 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. If an overlap o is specified, both systems can be delivered for an overlap of o time periods, starting in the first time period in which system i is delivered. This fulfills the **System Obviation** business rule.

$\forall (i, j, o) \in SysObviations, t, t^*$ where $i, j \in DeliverableSys$ and $t^* + o \leq t \leq T$

$$bSysDelivered_{j,t} <= 1 - bSysDelivered_{i,t^*} \tag{6.52}$$

- This constraint ensures that for each synchronization set s, the number of groups of synchronized systems in the synchronized mission for the synchronized component 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 $First(Z) = z_1$.

$\forall s, m \in SyncSetMissions_s, t \leq T$ where $m \neq First(SyncSetMissions_s)$

$$\sum_{\substack{i,m^*c: \\ (i,m^*)\in Roles \\ m^*=First(SyncSetMissions_s) \\ i\in SyncSetSys_s \\ c=SyncSetComponents_s}} NumGpInMission_{i,m^*,c,t} = \sum_{\substack{i,c: \\ (i,m,c)\in Roles \\ i\in SyncSetSys_s \\ c=SyncSetComponents_s}} NumGpInMission_{i,m,c,t} \tag{6.53}$$

- Constraints (6.54)–(6.56) properly set the value of the $bSysInStorageExchangeable_{i,c,t}$ indicator variable and then enforce the **Storage Consumption Priority** business rule.

$\forall i, c, t \leq T$ where $\exists (i,j) \in StorageUpgUsePriorities$

$$bSysInStorageExchangeable_{i,c,t} \leq NumSysInStorageExchangeable_{i,c,t} \tag{6.54}$$

$\forall i, c, t \leq T$ where $\exists (i,j) \in StorageUpgUsePriorities$

$$TotalSysPopulation * bSysInStorageExchangeable_{i,c,t} \geq NumSysInStorageExchangeable_{i,c,t} \tag{6.55}$$

$\forall i, c, t \leq T$ where $\exists (i,j) \in StorageUpgUsePriorities$

$$\sum_{\substack{j,j^*: \\ (i,j)\in StoragePriorityPair \\ (j,j^*)\in StorageUpg \\ t+UpgProdDelay_{j,j^*}\leq T}} iNumInStorageUpg_{j,j^*,c,t+UpgProdDelay_{j,j^*}} \leq (NumStorageUpg + 1) * TotalSysPopulation * (1 - bSysInStorageExchangeable_{i,c,t}) \tag{6.56}$$

- Constraints (6.57) – (6.59) properly set $bSysBeforePurchInStorageExchangeable_{i,c,t}$ and then enforce the **Upgrades Trump Purchases** business rule.

$\forall i \in StorageUpgBeforePurch, c, t \leq T$

$$bSysBeforePurchInStorageExchangeable_{i,c,t} \leq NumSysInStorageExchangeable_{i,c,t} \tag{6.57}$$

$\forall i \in StorageUpgBeforePurch, c, t \leq T$

$$TotalSysPopulation * bSysBeforePurchInStorageExchangeable_{i,c,t} \geq NumSysInStorageExchangeable_{i,c,t} \tag{6.58}$$

$$\forall i \in StorageUpgBeforePurch, c, t \leq T$$

$$\sum_{\substack{j: \\ (i,j) \in StorageUpg \\ j \in PurchasableSys \\ t+PurchProdDelay_j \leq T}} iNumBatchesPurch_{j,c,t+PurchProdDelay_j}$$

$$\leq TotalSysPopulation * (1 - bSysBeforePurchInStorageExchangeable_{i,c,t}) \tag{6.59}$$

- Constraint (6.60) enforces that each system i in component c is retired before exceeding its economic useful life for time t in the conventional time horizon. This fulfills the **Economic Useful Life** business rule.

$$\forall (i, m, c) \in Roles, t \leq \mathcal{T} \text{ where } t > SysUsefulLife_i - InitialRoleAge_{i,m,c} \leq \mathcal{T}$$

$$iSysYearAgeOverages_{i,m,c,t}$$

$$\geq InitialSysInMission_{i,m,c}$$

$$+ \begin{cases} SysFieldedToRoleSoFar_{i,m,c,t-SysUsefulLife_i} & \text{if } t > SysUsefulLife_i \\ 0 & \text{otherwise} \end{cases}$$

$$- SysRetiredFromRoleSoFar_{i,m,c,t} \tag{6.60}$$

Budget Constraints

- Constraints (6.61) – (6.63) combine to partially fulfill the **Component Earmarks** business rule for all components in the conventional and extended time horizons. The amount of each earmark spent in each component in each time period is captured in $fComponentEarmarkSpent_{c,t}$.

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

$$fComponentEarmarkSpent_{c,t} \leq ComponentEarmark_{c,t} \tag{6.61}$$

$$\forall c, t \leq T$$

$$fComponentEarmarkSpent_{c,t}$$

$$\leq ComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t}$$

$$+ FutureComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t} \tag{6.62}$$

$$\forall c, T < t \leq \mathcal{T}$$

$$fComponentEarmarkSpent_{c,t}$$

$$\leq FutureComponentPurchAndUpgExpenseWithoutPfEarmark_{c,t} \tag{6.63}$$

- Constraints (6.64) – (6.69) combine to partially fulfill the **Product Family Earmarks** business rule for all product families in the conventional and extended time horizons. The amount of each earmark spent in each product family in each time period for production is captured in $fPfEarmarkSpentProduction_{p,t}$ and for RDT&E is captured in $fPfEarmarkSpentRdte_{p,t}$.

$$\forall p, t \leq \mathcal{T}$$
$$fPfEarmarkSpentProduction_{p,t} + fPfEarmarkSpentRdte_{p,t} \leq PfEarmark_{p,t} \tag{6.64}$$

$$\forall p, t \leq T$$
$$fPfEarmarkSpentProduction_{p,t} \leq PfPurchAndUpgExpense_{p,t} + NonPurchAndUpgPfExpenses_{p,t} + FuturePfPurchAndUpgExpenses_{p,t} + FutureNonPurchAndUpgPfExpenses_{p,t} \tag{6.65}$$

$$\forall p, T < t \leq \mathcal{T}$$
$$fPfEarmarkSpentProduction_{p,t} \leq FuturePfPurchAndUpgExpenses_{p,t} + FutureNonPurchAndUpgPfExpenses_{p,t} \tag{6.66}$$

$$\forall p \in PfWithRdte, t \leq T$$
$$fPfEarmarkSpentRdte_{p,t} \leq RdteEffortExpense_{p,t} \tag{6.67}$$

$$\forall p \in PfWithRdteActive, T < t \leq \mathcal{T}$$
$$fPfEarmarkSpentRdte_{p,t} \leq bPfActive_{p,t} * PfRdteActiveCost_{p} \tag{6.68}$$

$$\forall p \notin PfWithRdte, t \leq \mathcal{T}$$
$$fPfEarmarkSpentRdte_{p,t} = 0 \tag{6.69}$$

- Constraints (6.70) and (6.73) partially fulfill the **Component Earmarks** business rule. Constraints (6.70), (6.72), and (6.73) partially fulfill the **Product Family Earmarks** business rule. Constraints (6.70) – (6.73) 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. In constraint (6.73), 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. Constraints (6.70) and (6.73) enforce the yearly budgets with earmarks.

$$\begin{aligned}
&\forall t \leq T \text{ where } ProcureBudget_t < \infty \\
&\quad ProcureExpense_t + FutureProcureExpense_t \\
&\qquad \leq ProcureBudget_t \\
&\qquad + CombinedComponentEarmarkSpent_t \\
&\qquad + CombinedPfEarmarkSpentProduction_t \\
&\qquad + fProcureBudgetOverrun_t
\end{aligned} \tag{6.70}$$

$$\begin{aligned}
&\forall t \leq T \text{ where } OSBudget_t < \infty \\
&\quad OSExpense_t + FutureOSExpense_t \\
&\qquad \leq OSBudget_t + fOSBudgetOverrun_t
\end{aligned} \tag{6.71}$$

$$\begin{aligned}
&\forall t \leq T \text{ where } RdteBudget_t < \infty \\
&\quad RdteExpense_t + FutureRdteExpense_t \\
&\qquad \leq RdteBudget_t \\
&\qquad + CombinedPfEarmarkSpentRdte_t \\
&\qquad + fRdteBudgetOverrun_t
\end{aligned} \tag{6.72}$$

$$\begin{aligned}
&\forall t \leq T \text{ where } CombinedBudget_t < \infty \\
&\quad b_{Proc} * (ProcureExpense_t + FutureProcureExpense_t) \\
&\qquad + b_{OS} * (OSExpense_t + FutureOSExpense_t) \\
&\qquad + b_{Rdte} * (RdteExpense_t + FutureRdteExpense_t) \\
&\qquad \leq CombinedBudget_t \\
&\qquad + b_{Proc} * (CombinedComponentEarmarkSpent_t \\
&\qquad + CombinedPfEarmarkSpentProduction_t) \\
&\qquad + b_{Rdte} * CombinedPfEarmarkSpentRdte_t \\
&\qquad + fCombinedBudgetOverrun_t
\end{aligned} \tag{6.73}$$

- Constraints (6.74) and (6.77) partially fulfill the **Component Earmarks** business rule. Constraints (6.74), (6.76), and (6.77) partially fulfill the **Product Family Earmarks** business rule. Constraints (6.74) – (6.77) 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.

$$\forall\ T < t \leq \mathcal{T} \text{ where } FutureProcureBudget_t < \infty$$
$$FutureProcureExpense_t$$
$$\leq FutureProcureBudget_t$$
$$+ CombinedComponentEarmarkSpent_t$$
$$+ CombinedPfEarmarkSpentProduction_t$$
$$+ fFutureProcureBudgetOverrun_t \tag{6.74}$$

$$\forall\ T < t \leq \mathcal{T} \text{ where } FutureOSBudget_t < \infty$$
$$FutureOSExpense_t$$
$$\leq FutureOSBudget_t + fFutureOSBudgetOverrun_t \tag{6.75}$$

$$\forall\ T < t \leq \mathcal{T} \text{ where } FutureRdteBudget_t < \infty$$
$$FutureRdteExpense_t$$
$$\leq FutureRdteBudget_t + fFutureRdteBudgetOverrun_t \tag{6.76}$$

$$\forall\ T < t \leq \mathcal{T} \text{ where } CombinedBudget_t < \infty$$
$$b_{Proc} * FutureProcureExpense_t$$
$$+ b_{OS} * FutureOSExpense_t$$
$$+ b_{Rdte} * FutureRdteExpense_t$$
$$\leq FutureCombinedBudget_t$$
$$+ b_{Proc} * (CombinedComponentEarmarkSpent_t$$
$$+ CombinedPfEarmarkSpentProduction_t)$$
$$+ b_{Rdte} * CombinedPfEarmarkSpentRdte_t$$
$$+ fFutureCombinedBudgetOverrun_t \tag{6.77}$$

- Constraints (6.78) – (6.81) 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 and non-future system expenses across both the conventional and extended time horizons.

  if $TotalProcureBudget < \infty$

  $$\sum_{t \leq T} ProcureExpense_t + \sum_{t \leq \mathcal{T}} FutureProcureExpense_t$$
  $$\leq TotalProcureBudget + fTotalProcureBudgetOverrun \tag{6.78}$$

  if $TotalOSBudget < \infty$

  $$\sum_{t \leq T} OSExpense_t + \sum_{t \leq \mathcal{T}} FutureOSExpense_t$$
  $$\leq TotalOSBudget + fTotalOSBudgetOverrun \tag{6.79}$$

if $TotalRdteBudget < \infty$

$$\sum_{t \leq T} RdteExpense_t + \sum_{t \leq \mathcal{T}} FutureRdteExpense_t \leq TotalRdteBudget + fTotalRdteBudgetOverrun \tag{6.80}$$

if $TotalCombinedBudget < \infty$

$$\sum_{t \leq T} CombinedExpense_t + \sum_{t \leq \mathcal{T}} FutureCombinedExpense_t \leq TotalCombinedBudget + fTotalCombinedBudgetOverrun \tag{6.81}$$

Group Density Levels

- Constraints (6.82) and (6.83) 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 for any component c and those transitions achieve a density level of $\ell \in UpgDensityLevels_m$ groups. These constraints partially fulfill the **Minimum Group Transition Density** business rule.

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

$$\sum_{\substack{j,c,t: \\ (j,i,m,c) \in Transitions \\ t \leq T}} NumGpTransit_{j,i,m,c,t} \geq \sum_{\ell \in UpgDensityLevels_m} (bTransitionedToDensityLevel_{i,m,\ell} * \ell) \tag{6.82}$$

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

$$\begin{aligned} &\sum_{\substack{j,c,t: \\ (j,i,m,c) \in Transitions \\ t \leq T}} NumGpTransit_{j,i,m,c,t} \\ &\leq \sum_{\substack{\ell: \\ \ell \in UpgDensityLevels_m \\ \ell \neq \max(UpgDensityLevels_m)}} (bTransitionedToDensityLevel_{i,m,\ell} * \ell) \\ &+ \sum_{\substack{c,\ell: \\ \ell \in UpgDensityLevels_m \\ \ell = \max(UpgDensityLevels_m)}} (bTransitionedToDensityLevel_{i,m,\ell} * GpPerComponentMission_{c,m}) \end{aligned} \tag{6.83}$$

- Constraint (6.84) ensures that system i in mission m across all components c 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 (6.82) and (6.83), this fulfills the **Minimum Group Transition Density** business rule.

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

$$\sum_{\ell \in UpgDensityLevels_m} bTransitionedToDensityLevel_{i,m,\ell} \leq 1 \tag{6.84}$$

- Constraints (6.85) and (6.86) ensure the $bHasFinalDensity_{i,m,\ell}$ indicator variable equals 1 if and only if the cumulative number of groups of system i in mission m for all components has density $\ell \in FinalDensityLevels_m$ groups at time $\mathcal{T}$. Note that final density constraints only apply to system types which are fielded to but never retired from a given role. These constraints partially fulfill the **Minimum Group Final Density** business rule.

$\forall(i,m,\ell) \in FinalDensityFlags$

$$\begin{aligned}
&\sum_{\substack{c: \\ (i,m,c)\in Roles}} NumGpInMission_{i,m,c,\mathcal{T}} \\
&\quad \geq (bHasFinalDensity_{i,m,\ell} + TransitionedToRole_{i,m} - 1) * \ell \\
&\quad - \begin{cases} \displaystyle\sum_{\substack{c: \\ (i,m,c)\in Roles}} GpRetiredFromRoleSoFar_{i,m,c,\mathcal{T}} * \ell & \text{if } \ell = \max(FinalDensityLevels_m) \\ 0 & \text{otherwise} \end{cases}
\end{aligned} \tag{6.85}$$

$\forall(i,m)$ where $\exists(i,m,\ell) \in FinalDensityFlags$

$$\begin{aligned}
&\sum_{\substack{c: \\ (i,m,c)\in Roles}} NumGpInMission_{i,m,c,\mathcal{T}} \\
&\quad \leq \sum_{\substack{\ell: \\ \ell \in FinalDensityLevels_m \\ \ell \neq \max(FinalDensityLevels_m)}} (bHasFinalDensity_{i,m,\ell} * \ell) \\
&\quad + \sum_{\substack{c,\ell: \\ \ell \in FinalDensityLevels_m \\ \ell = \max(FinalDensityLevels_m)}} (bHasFinalDensity_{i,m,\ell} * GpPerComponentMission_{c,m})
\end{aligned} \tag{6.86}$$

- Constraint (6.87) ensures that system i in mission m across all components can satisfy at most 1 of the final transition density levels. Note that if system i never transitions into system m for any component at any time, then all three of the $bHasFinalDensity$ binaries will be 0. Together with (6.85) and (6.86), this fulfills the **Minimum Group Final Density** business rule.

$\forall(i,m)$ where $\exists(i,m,\ell) \in FinalDensityFlags$

$$\sum_{\ell \in FinalDensityLevels_m} bHasFinalDensity_{i,m,\ell} \leq 1 \tag{6.87}$$

System Production Constraints

- Constraints (6.88) and (6.89) ensure that the $bSysDeliveredComponent_{i,c,t}$ flag is 1 if and only if at least one of system i is delivered to component c in time period t. This flag is then used to help fulfill the **Delivery Implies Fielding** business rule. It is also used to set the $bSysDelivered_{i,t}$ flag.

$$\forall i \in DeliverableSys, c, t \leq T$$
$$bSysDeliveredComponent_{i,c,t} \leq NumSysDeliveredComponent_{i,c,t} \tag{6.88}$$

$$\forall i \in DeliverableSys, c, t \leq T$$
$$TotalSysPopulation * bSysDeliveredComponent_{i,c,t} \geq NumSysDeliveredComponent_{i,c,t} \tag{6.89}$$

- Constraints (6.90)-(6.92) ensure that the $bSysHasDeliveredComponent_{i,c,t}$ flag is 1 if and only if at least one of system i has been delivered to component c at or before time period t. This flag is then used to help fulfill the **Fielding New Systems from Storage** business rule.

$$\forall i \in DeliverableSys, c, t \leq T$$
$$bSysHasDeliveredComponent_{i,c,t} \geq bSysDeliveredComponent_{i,c,t} \tag{6.90}$$

$$\forall i \in DeliverableSys, c, t \leq T$$
$$bSysHasDeliveredComponent_{i,c,t} \leq \sum_{t^* \leq t} bSysDeliveredComponent_{i,c,t^*} \tag{6.91}$$

$$\forall i \in DeliverableSys, c, t \leq T-1$$
$$bSysHasDeliveredComponent_{i,c,t} \leq bSysHasDeliveredComponent_{i,c,t+1} \tag{6.92}$$

- Constraints (6.93) and (6.94) ensure that the $bSysDelivered_{i,t}$ flag is 1 if and only at least one of system i is delivered in time period t. This flag is then used to help fulfill a variety of business rules.

$$\forall i \in DeliverableSys, c, t \leq T$$
$$bSysDelivered_{i,t} \geq bSysDeliveredComponent_{i,c,t} \tag{6.93}$$

$$\forall i \in DeliverableSys, t \leq T$$
$$bSysDelivered_{i,t} \leq \sum_{c} bSysDeliveredComponent_{i,c,t} \tag{6.94}$$

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

  $\forall i, c, t \leq T$ where $i \in DeliverableSys$ and $i \notin FreeInterimUpgSys$

  $$\sum_{\substack{j,m,t^*: \\ (j,i,m,c) \in Transitions \\ t \leq t^* \leq T}} NumGpTransit_{j,i,m,c,t^*} \geq bSysDeliveredComponent_{i,c,t} \tag{6.95}$$

- Constraints (6.96) and (6.97) ensure that the $bSysEverDelivered_i$ 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.

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

  $$bSysEverDelivered_i \geq bSysDelivered_{i,t} \tag{6.96}$$

  $\forall i \in DeliverableSys$

  $$bSysEverDelivered_i \leq \sum_{t \leq T} bSysDelivered_{i,t} \tag{6.97}$$

Product Family Constraints

- Constraints (6.98) and (6.99) ensure that for each product family having a minimum production rate, the $bPfDelivered_{p,t}$ 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**, **Production Smoothing**, and **Product Family Obviation** business rules.

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

  $$bPfDelivered_{p,t} \leq \sum_{i \in ProductFamily_p \cap DeliverableSys} bSysDelivered_{i,t} \tag{6.98}$$

  $\forall p \in PfWithProdCtrls, i \in ProductFamily_p \cap DeliverableSys, t \leq T$

  $$bPfDelivered_{p,t} \geq bSysDelivered_{i,t} \tag{6.99}$$

- Constraints (6.100) and (6.101) ensure that for each product family having a minimum cumulative production, the $bPfEverDelivered_p$ indicator variable is 1 if and only if at least one of the member systems of p is ever delivered. This flag helps support the **Family Minimum Cumulative Delivery** business rule.

  $\forall p \in PfWithProdCtrls$

  $$bPfEverDelivered_p \leq \sum_{t \leq T} bPfDelivered_{p,t} \tag{6.100}$$

$$\forall p \in PfWithProdCtrls, t \leq T$$
$$bPfEverDelivered_p \geq bPfDelivered_{p,t} \tag{6.101}$$

- Constraints (6.102) and (6.103) ensure that for each product family p having an active cost and for each t, the $bPfActive_{p,t}$ 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.

$$\forall p \in PfWithAnyActive, t \leq \mathcal{T}$$
$$bPfActive_{p,t} \leq \sum_{i \in ProductFamily_p} (NumSysInProduction_{i,t} + NumSysInAdminPeriod_{i,t}) + \begin{cases} \sum_{i \in ProductFamily_p} NumLripSysActive_{i,t} & t \leq T \\ 0 & t > T \end{cases} \tag{6.102}$$

$$\forall p \in PfWithAnyActive, t \leq \mathcal{T}$$
$$10 * TotalSysPopulation * bPfActive_{p,t} \geq \sum_{i \in ProductFamily_p} (NumSysInProduction_{p,t} + NumSysInAdminPeriod_{i,t}) + \begin{cases} \sum_{i \in ProductFamily_p} NumLripSysActive_{i,t} & t \leq T \\ 0 & t > T \end{cases} \tag{6.103}$$

- Constraints (6.104)–(6.106) ensure that for each product family p having a start-up cost profile, the flag $bPfStartup_{p,t}$ 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.

$$\forall p \in PfWithStartup, t \leq T$$
$$bPfStartup_{p,t} \leq \sum_{i \in ProductFamily_p} (NumSysInProduction_{i,t} + NumSysInAdminPeriod_{i,t}) \tag{6.104}$$

$$\forall p \in PfWithStartup, t \leq T$$
$$10 * TotalSysPopulation * \sum_{t^* \leq t} bPfStartup_{p,t^*} \geq \sum_{i \in ProductFamily_p} (NumSysInProduction_{i,t} + NumSysInAdminPeriod_{i,t}) \tag{6.105}$$

$\forall p \in PfWithStartup$

$$\sum_{t \leq T} bPfStartup_{p,t} \leq 1 \tag{6.106}$$

- Constraints (6.107)–(6.109) ensure that for each product family p having LRIP, the $bPfFrpStarted_{p,t}$ 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.

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

$$bPfFrpStarted_{p,t} \leq \sum_{i \in ProductFamily_p \cap DeliverableSys} bSysDelivered_{i,t} \tag{6.107}$$

$\forall p, i, t \leq T$ where $p \in PfWithLrip$ and $i \in ProductFamily_p \cap DeliverableSys$

$$\sum_{t^* \leq t} bPfFrpStarted_{p,t^*} \geq bSysDelivered_{i,t} \tag{6.108}$$

$\forall p \in PfWithLrip$

$$\sum_{t \leq T} bPfFrpStarted_{p,t} \leq 1 \tag{6.109}$$

- Constraints (6.110) – (6.112) ensure that for each product family p with a fielding ratio, the $bPfFirstYearFielding_{p,t}$ flag is 1 if and only if time t is the first time that a member system of p is fielded to any component. This helps fulfill the **Product Family Ratios** business rule.

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

$$bPfFirstYearFielding_{p,t} \leq \sum_{c} PfSysFielded_{p,c,t} \tag{6.110}$$

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

$$TotalSysPopulation * \sum_{t^* \leq t} bPfFirstYearFielding_{p,t^*} \geq \sum_{c} PfSysFielded_{p,c,t} \tag{6.111}$$

$\forall p \in PfWithRatios$

$$\sum_{t} bPfFirstYearFielding_{p,t} \leq 1 \tag{6.112}$$

- Constraints (6.113) – (6.115) ensure that for each product family with a fielding ratio, the $bPfLastYearFielding_{p,c,t}$ flag is 1 if and only if time t is the last time that a member system of p is fielded to component c. This helps fulfill the **Product Family Ratios** business rule.

$$\forall p \in PfWithRatios, c, t \leq T$$
$$bPfLastYearFielding_{p,c,t} \leq PfSysFielded_{p,c,t} \tag{6.113}$$

$$\forall p \in PfWithRatios, c, t \leq T$$
$$TotalSysPopulation * \sum_{t^* \geq t} bPfLastYearFielding_{p,c,t^*} \geq \sum_{c} PfSysFielded_{p,c,t} \tag{6.114}$$

$$\forall p \in PfWithRatios, c$$
$$\sum_{t} bPfLastYearFielding_{p,c,t} \leq 1 \tag{6.115}$$

- Constraint (6.116) ensures that for product family p with ratios, if any systems from this product family begin fielding to one component, then systems must begin fielding to all other components. This helps fulfill the **Product Family Ratios** business rule.

$$\forall p \in PfWithRatios, c$$
$$\sum_{t} bPfFirstYearFielding_{p,t} \leq \sum_{t} bPfLastYearFielding_{p,c,t} \tag{6.116}$$

- Constraints (6.117) – (6.119) ensure that for each product family with a fielding ratio, the $bPfEnforceRatio_{p,c,t}$ flag is 1 if and only if the ratio for product family p is enforced for component c in time t. This helps fulfill the **Product Family Ratios** business rule.

$$\forall p \in PfWithRatios, c, t \leq T$$
$$\sum_{t^* \leq t} bPfFirstYearFielding_{p,t^*} + \sum_{t^* > t + PfRatioWindow_p - 1} bPfLastYearFielding_{p,c,t^*} - 1 \leq bPfEnforceRatio_{p,c,t} \tag{6.117}$$

$$\forall p \in PfWithRatios, c, t \leq T$$
$$\sum_{t^* \leq t} bPfFirstYearFielding_{p,t^*} \geq bPfEnforceRatio_{p,c,t} \tag{6.118}$$

$$\forall p \in PfWithRatios, c, t \leq T$$
$$\sum_{t^* > t + PfRatioWindow_p - 1} bPfLastYearFielding_{p,c,t^*} \geq bPfEnforceRatio_{p,c,t} \tag{6.119}$$

- Constraints (6.120) – (6.122) ensure that fielding to product family p is within the variance of the ratios for each component and time t. The partially fulfills the **Product Family Ratios** business rule.

$$\forall p, c_1, c_2, t \leq T - PfRatioWindow_p + 1$$
$$\text{where } PfComponentRatios_{p,c_1} \geq PfComponentRatios_{p,c_2} > 0$$
$$PfSysFieldedWindow_{p,c_1,t}$$
$$\leq \frac{PfComponentRatios_{p,c_1}}{PfComponentRatios_{p,c_2}} * (1 + PfRatioVariance_p)$$
$$* PfSysFieldedWindow_{p,c_2,t}$$
$$+ TotalSysPopulation * (1 - bPfEnforceRatio_{p,c_2,t}) \tag{6.120}$$

$$\forall p, c_1, c_2, t \leq T - PfRatioWindow_p + 1$$
$$\text{where } PfComponentRatios_{p,c_1} \geq PfComponentRatios_{p,c_2} > 0$$
$$PfSysFieldedWindow_{p,c_1,t}$$
$$\geq \frac{PfComponentRatios_{p,c_1}}{PfComponentRatios_{p,c_2}} * (1 - PfRatioVariance_p)$$
$$* PfSysFieldedWindow_{p,c_2,t}$$
$$- \frac{PfComponentRatios_{p,c_1}}{PfComponentRatios_{p,c_2}} * (1 - PfRatioVariance_p)$$
$$* TotalSysPopulation * (1 - bPfEnforceRatio_{p,c_1,t}) \tag{6.121}$$

$$\forall p \in PfWithRatios, c \text{ where } PfComponentRatios_{p,c} = 0$$
$$\sum_{t \leq T} PfSysFielded_{p,c,t} = 0 \tag{6.122}$$

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

$$\forall p, t, t^* \text{ where } PfAllowGaps_p = 0 \text{ and } t^* + 1 < t \leq T$$
$$bPfDelivered_{p,t} - bPfDelivered_{p,t-1} + bPfDelivered_{p,t^*} \leq 1 \tag{6.123}$$

- Constraints (6.124) and (6.125) 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.

$$\forall p \in PfWithStartup, t \leq T \text{ where } \sum_{t \leq t^* < T} PfStartupCostSchedule_{p,-t^*} > 0$$

$$bPfStartup_{p,t} = 0 \tag{6.124}$$

$$\forall p \in PfWithStartup, t \leq T \text{ where } \sum_{T-t < t^* < T} PfStartupCostSchedule_{p,t^*} > 0$$

$$bPfStartup_{p,t} = 0 \tag{6.125}$$

- This constraint ensures that the number of systems delivered by each product family to all components is greater than or equal to the specified requirment for that product family for each time period. This fulfills the **Family Minimum Per-Period Delivery** business rule.

$$\forall p, t \leq T \text{ where } PfDeliveryMin_{p,t} > 0$$

$$\sum_{\substack{i: \\ i \in ProductFamily_p}} NumSysDelivered_{i,t} + iPfDeliveryDeficit_{p,t} \geq PfDeliveryMin_{p,t} \tag{6.126}$$

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

$$\forall p, t \leq T \text{ where } PfDeliveryMax_{p,t} < \infty$$

$$\sum_{i \in ProductFamily_p} NumSysDelivered_{i,t} \leq PfDeliveryMax_{p,t} \tag{6.127}$$

- This constraint ensures that the cumulative number of systems delivered by each product family to all components is more than the specified capacity if systems are ever delivered to the product family. This fulfills this **Family Minimum Cumulative Delivery** business rule.

$$\forall p \text{ where } PfTotalDeliveryMin_p > 0$$

$$\sum_{\substack{i,t: \\ i \in ProductFamily_p \\ t \leq T}} NumSysDeliveredIncludingLripProduced_{i,t} + \sum_{\substack{i,t: \\ i \in ProductFamily_p \\ i \in CoastableSys \\ T < t \leq \mathcal{T}}} NumCoastingSysDelivered_{i,t} \geq bPfEverDelivered_p * PfTotalDeliveryMin_p \tag{6.128}$$

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

$\forall p$ where $PfTotalDeliveryMax_p < \infty$

$$\sum_{\substack{i,t: \\ i \in ProductFamily_p \\ t \leq T}} NumSysDeliveredIncludingLripProduced_{i,t}$$

$$+ \sum_{\substack{i,t: \\ i \in ProductFamily_p \\ i \in CoastableSys \\ T < t \leq \mathcal{T}}} NumCoastingSysDelivered_{i,t}$$

$$\leq PfTotalDeliveryMax_p \tag{6.129}$$

- This constraint ensures that if systems are delivered from product family p to any components 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.

$\forall p, t < T$ where $1 < PfMsr_p < PfDeliveryMax_{p,t}$

$$\sum_{i \in ProductFamily_p} NumSysDelivered_{i,t}$$

$$\geq bPfDelivered_{p,t} * PfMsr_p$$

$$- (1 - bPfDelivered_{p,t+1}) * TotalSysPopulation \tag{6.130}$$

- Constraint (6.131) partially fulfills the **RDT&E Costs** business rule by enforcing that an RDT&E delay is disallowed if any of the associated costs extend into future time periods.

$\forall p \in PfWithRdteDelays, d \in AllowedRdteDelays_p$

where $HasRdteInFutureTimePeriod_{p,d} = 1$

$$bRdteDelay_{p,d} = 0 \tag{6.131}$$

- Constraints (6.132)–(6.135) 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 $EnableRdteDelays = 1$

$\forall p \in PfWithRdteDelays, d$

$$bRdteDelay_{p,d} \leq \sum_{\substack{i: \\ i \in ProductFamily_p \cap DeliverableSys \\ FirstAvailable_i + d \leq T}} bSysDelivered_{i,FirstAvailable_i + d} \tag{6.132}$$

$\forall p \in PfWithRdteDelays, d, i$ where $i \in ProductFamily_p \cap DeliverableSys$
and $FirstAvailable_i + d \leq T$

$$\sum_{d^* \leq d} bRdteDelay_{p,d^*} \geq bSysDelivered_{i,FirstAvailable_i+d} \tag{6.133}$$

$\forall p \in PfWithRdteDelays$

$$\sum_{d} bRdteDelay_{p,d} \leq 1 \tag{6.134}$$

$\forall p \in PfWithRdteDelays, d \notin AllowedRdteDelays_p$

$$\sum_{d} bRdteDelay_{p,d} = 0 \tag{6.135}$$

- Constraints (6.136)–(6.138) 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 $EnableRdteDelays = 0$

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

$$bRdteDelay_{p,d} = 0 \tag{6.136}$$

$\forall p \in PfWithRdteDelays$

$$bRdteDelay_{p,0} \leq \sum_{\substack{i,t: \\ i \in ProductFamily_p \cap DeliverableSys \\ t \leq T}} bSysDelivered_{i,t} \tag{6.137}$$

$\forall p \in PfWithRdteDelays, i, t \leq T$ where $i \in ProductFamily_p \cap DeliverableSys$

$$bRdteDelay_{p,0} \geq bSysDelivered_{i,t} \tag{6.138}$$

LRIP Constraints

- Constraints (6.139)–(6.141) ensure that the $bLripSysBaseYear_{p,i,t}$ 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.

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$bLripSysBaseYear_{p,i,t} \leq bPfFrpStarted_{p,t} \tag{6.139}$$

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$bLripSysBaseYear_{p,i,t} \leq bSysEverDelivered_{i} \tag{6.140}$$

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$bLripSysBaseYear_{p,i,t} \geq bSysEverDelivered_{i} + bPfFrpStarted_{p,t} - 1 \tag{6.141}$$

- 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.

$$\forall (p,i) \in LripProfiles, t^* \in LripYears, t \text{ where } LripPreProduction_{p,i,t^*} > 0$$
$$\text{and } t <= t^* + LripDelay_{p,i} + \psi_{p,i}$$
$$bLripSysBaseYear_{p,i,t} = 0 \tag{6.142}$$

- Constraint (6.143) ensures that for all systems i in product family p with an LRIP profile that the total number of LRIP systems that finish production in time t, denoted by $LripProduced_{p,i,t}$, are partitioned into components chosen by the optimization. This partially satisfies the **LRIP Profiles** business rule.

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$\sum_{c} iLripProduced_{c,p,i,t} = LripProduced_{p,i,t} \tag{6.143}$$

- Constraint (6.144) ensures that for all systems i in product family p with an LRIP profile that the total number of LRIP systems delivered in time t, denoted by $LripDelivered_{p,i,t}$, are partitioned into components chosen by the optimization. This partially satisfies the **LRIP Profiles** business rule.

$$\forall (p,i) \in LripProfiles, t \leq T$$
$$\sum_{c} iLripDelivered_{c,p,i,t} = LripDelivered_{p,i,t} \tag{6.144}$$

- Constraint (6.145) ensures that all consumed seed systems i, denoted by $LripSeedsConsumed_{i,t}$, are partitioned into components chosen by the optimization. This partially satisfies the **LRIP Profiles** business rule.

$$\forall i, t \leq T$$
$$\sum_{c} iLripSeedsConsumed_{c,i,t} = LripSeedsConsumed_{i,t} \tag{6.145}$$

- Constraint (6.146) ensures that for all systems i in product family p with an LRIP profile that the total number of LRIP systems delivered in time t to component c cannot exceed the total number of LRIP systems that have completed production for component c in time t. This partially satisfies the **LRIP Profiles** business rule.

$$\forall c, (p,i) \in LripProfiles, t \leq T$$
$$iLripDelivered_{c,p,i,t} <= iLripProduced_{c,p,i,t} \tag{6.146}$$

Production Smoothing Constraints

- These constraints ensure that the total number of systems i produced for any component 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.

$$\forall p, t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p < t \leq T$$
$$\sum_{i \in ProductFamily_p} NumSysDelivered_{i,t}$$
$$\leq (1 + 0.5 * MaxDeliveryVariance_p) * fMedianDeliveryLevel_p \tag{6.147}$$

$$\forall p, t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p < t < T - 1$$
$$\sum_{i \in ProductFamily_p} NumSysDelivered_{i,t}$$
$$\geq (1 - 0.5 * MaxDeliveryVariance_p) * fMedianDeliveryLevel_p$$
$$- \sum_{t^* \in \{0,\ldots,RampUp_p\}} (1 - bPfDelivered_{p,t-t^*}) * TotalSysPopulation$$
$$- (1 - bPfDelivered_{p,t+1}) * TotalSysPopulation \tag{6.148}$$

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

$$\forall p, t \text{ where } MaxDeliveryVariance_p \geq 0 \text{ and } RampUp_p > 0 \text{ and } 1 < t \leq T$$
$$\sum_{i \in ProductFamily_p} NumSysDelivered_{i,t}$$
$$\geq \sum_{i \in ProductFamily_p} NumSysDelivered_{i,t-1}$$
$$- \begin{cases} 0 & \text{if } t \leq RampUp_p + 1 \\ bPfDelivered_{p,t-RampUp_p-1} * TotalSysPopulation & \text{if } t > RampUp_p + 1 \end{cases}$$
$$- (1 - bPfDelivered_{p,t}) * TotalSysPopulation \tag{6.149}$$

- This constraint ensures for product family obviation pairs (p, q) that once any system from p is delivered, systems from q can no longer be delivered. In other words, deliveries from q can only occur before deliveries from p. This fulfills the **Product Family Obviation** business rule.

$$\forall (p,q) \in PfObviations, t, t^* \text{ where } t^* \leq t \leq T$$
$$bPfDelivered_{p,t^*} <= 1 - bPfDelivered_{q,t} \tag{6.150}$$

Coasting Systems Constraints

- Constraints (6.151) – (6.152) set the coasting level of coasting system i in mission m for component c in all time periods t that coasting is enforced. This partially fulfills the **Coasting System Fielding** business rule.

$$\begin{aligned}&\forall (i,m,c) \in CoastableRoles, T < t < \mathcal{T} \\ &- GpPerComponentMission_{c,m} * (2 - bIsCoasting_{i,m,c,t} - bIsCoasting_{i,m,c,t+1}) \\ &\quad + iCoastingLevel_{i,m,c} \\ &\quad \leq \sum_{j:(j,i,m,c)\in CoastablePurchaseTransitions} iNumGpCoastingPurch_{j,i,m,c,t} \\ &\quad + \sum_{j:(j,i,m,c)\in CoastableUpgradeTransitions} iNumGpCoastingUpg_{j,i,m,c,t}\end{aligned} \tag{6.151}$$

$$\begin{aligned}&\forall (i,m,c) \in CoastableRoles, T < t \leq \mathcal{T} \\ &GpPerComponentMission_{c,m} * (1 - bIsCoasting_{i,m,c,t}) \\ &\quad + iCoastingLevel_{i,m,c} \\ &\quad \geq \sum_{j:(j,i,m,c)\in CoastablePurchaseTransitions} iNumGpCoastingPurch_{j,i,m,c,t} \\ &\quad + \sum_{j:(j,i,m,c)\in CoastableUpgradeTransitions} iNumGpCoastingUpg_{j,i,m,c,t}\end{aligned} \tag{6.152}$$

- Constraints (6.153) – (6.154) ensure that the number of coasting systems produced in the last conventional time frame is within half a purchase batch size so all coasting systems have the opportunity to coast in future time periods. This partially fulfills the **Coasting System Fielding** business rule.

$$\begin{aligned}&\forall i \in CoastableSys \\ &NumSysDelivered_{i,T} - 0.5 * PurchBatchSize_i \\ &\quad \leq \sum_{m,c:(i,m,c)\in CoastableRoles} iCoastingLevel_{i,m,c} * SysPerGp_m\end{aligned} \tag{6.153}$$

$$\forall i \in CoastableSys$$
$$NumSysDelivered_{i,T} + 0.5 * PurchBatchSize_i \geq \sum_{m,c:(i,m,c)\in CoastableRoles} iCoastingLevel_{i,m,c} * SysPerGp_m \quad (6.154)$$

- Constraint (6.155) ensures system type i in any mission m and any component c can only coast if system type i was delivered in time T to any mission or component. This partially fulfills the **Coasting System Fielding** business rule.

$$\forall (i,m,c) \in CoastableRoles, T < t \leq \mathcal{T}$$
$$bIsCoasting_{i,m,c,t} \leq NumSysDelivered_{i,T} \quad (6.155)$$

- Constraint (6.156) ensures that once there is only coasting system type i in mission m in component c then we cease coasting of system i to mission m in component c. This partially fulfills the **Coasting System Fielding** business rule.

$$\forall (i,m,c) \in CoastableRoles, T < t \leq \mathcal{T}$$
$$GpPerComponentMission_{c,m} * bIsCoasting_{i,m,c,t} \geq \sum_{j:(j,i,m,c)\in CoastablePurchTransitions} iNumGpCoastingPurch_{j,i,m,c,t} + \sum_{j:(j,i,m,c)\in CoastableUpgTransitions} iNumGpCoastingUpg_{j,i,m,c,t} \quad (6.156)$$

- Constraint (6.157) ensures that once coasting system type i ceases production and fielding to mission m in component c then it cannot restart coasting. This partially fulfills the **Coasting System Fielding** business rule.

$$\forall (i,m,c) \in CoastableRoles, T < t < \mathcal{T}$$
$$bIsCoasting_{i,m,c,t} \geq bIsCoasting_{i,m,c,t+1} \quad (6.157)$$

- Constraints (6.158) – (6.159) ensure that production for coasting systems are not allowed to violate product family production minimums or variance bands in the the final conventional time T. This partially fulfills the **Coasting System Fielding** business rule.

$$\forall p, (i,m,c) \in CoastableRoles, T < t \leq \mathcal{T} \text{ where } 1 < PfMsr_p \leq PfDeliveryMax_{p,T} \text{ and } i \in ProductFamily_p$$
$$PfMsr_p * bIsCoasting_{i,m,c,t} \leq \sum_{i^*\in ProductFamily_p} NumSysDelivered_{i^*,T} \quad (6.158)$$

$$\forall p, (i,m,c) \in CoastableRoles, T < t \leq \mathcal{T} \text{ where } 0 < MaxDeliveryVariance_p$$
$$\text{and } i \in ProductFamily_p$$
$$(1 - 0.5 * MaxDeliveryVariance_p) * fMedianDeliveryLevel_p$$
$$\leq \sum_{i^* \in ProductFamily_p} NumSysDelivered_{i^*,T}$$
$$+ TotalSysPopulation * (1 - bIsCoasting_{i,m,c,t}) \tag{6.159}$$

- Constraint (6.160) ensures that the number of groups of system type i in mission m in component c for all time t is non-negative. This partially fulfills the **Coasting System Fielding** and the **Constant System Populations** business rules.

$$\forall (i,m,c) \in Roles, t$$
$$NumGpInMission_{i,m,c,t} \geq 0 \tag{6.160}$$

Future Program Constraints

- Constraints (6.161) and (6.162) ensure that each future program $\mathcal{F}$ can be activated if and only if at least one of the future systems in any component associated with that program is also activated. These fulfill the first part of the **Future Program Activation** business rule.

$$\forall \mathcal{F}$$
$$\sum_{\substack{c \\ \mathcal{J} \in FutureProgram_{\mathcal{F}}}} bFutureSysComponent_{\mathcal{J},c}$$
$$\leq NumFutureSys * NumComponents * bFutureProgram_{\mathcal{F}} \tag{6.161}$$

$$\forall \mathcal{F}$$
$$\sum_{\substack{c \\ \mathcal{J} \in FutureProgram_{\mathcal{F}}}} bFutureSysComponent_{\mathcal{J},c} \geq bFutureProgram_{\mathcal{F}} \tag{6.162}$$

- Constraints (6.163) and (6.164) ensure that each future system $\mathcal{J}$ is activated in component c if and only if at least one group of non-future systems is replaced by $\mathcal{J}$ in component c at some time t within the conventional or extended horizon. These partially fulfill the part of the **Future System Fielding** business rule.

$$\forall \mathcal{J}, c$$
$$\sum_{\substack{i,m,t: \\ (i,\mathcal{J},m,c) \in FutureTransitions \\ t \leq \mathcal{T}}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \leq 10,000 * bFutureSysComponent_{\mathcal{J},c} \tag{6.163}$$

$$\forall \mathcal{J}, c$$

$$\sum_{\substack{i,m,t: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ t\leq \mathcal{T}}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \geq bFutureSysComponent_{\mathcal{J},c} \tag{6.164}$$

- Some future programs may only be activated if every associated future system is fielded in every component. 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.

$$\forall \mathcal{F}, \mathcal{J} \in FutureProgram_{\mathcal{F}}, c \text{ where } FieldAllWithinProgram_{\mathcal{F}} = 1$$
$$bFutureProgram_{\mathcal{F}} \leq bFutureSysComponent_{\mathcal{J},c} \tag{6.165}$$

- Constraints (6.166) and (6.167) ensure that once future systems start fielding to a mission in a component, non-future systems are no longer allowed to field to that mission in the component, neither by mission upgrades nor storage swaps. These fulfill the **Future Obviates Present** business rule.

$$\forall \mathcal{J}, c$$
$$10,000 * (1 - bFutureSysComponent_{\mathcal{J},c}) \geq \sum_{\substack{i,j,m,t: \\ (i,j,m,c)\in Transitions \\ m=FutureMissionMap_{\mathcal{J}} \\ FutureSysFirstYearFielding_{\mathcal{J}}\leq t\leq \mathcal{T}}} iNumGpFromStorage_{i,j,m,c,t} \tag{6.166}$$

$$\forall \mathcal{J}, c$$
$$10,000 * (1 - bFutureSysComponent_{\mathcal{J},c}) \geq \sum_{\substack{i,j,m,t: \\ (i,j,m,c)\in Transitions \\ m=FutureMissionMap_{\mathcal{J}} \\ FutureSysFirstYearFielding_{\mathcal{J}}\leq t\leq \mathcal{T}}} iNumGpInMissionUpg_{i,j,m,c,t} \tag{6.167}$$

- This constraint ensures that if future system $\mathcal{J}$ is activated, then it must fielded according to a fixed schedule given by the input $FutureSysFieldingProfile_{\mathcal{J},c,t}$. Note that the optimization can still decided which non-future systems will be replaced first. Together with (6.163) and (6.164), this satisfies the **Future System Fielding** business rule.

$$\forall \mathcal{J}, c, t \leq \mathcal{T}$$
$$\sum_{\substack{i,m: \\ (i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} = bFutureSysComponent_{\mathcal{J},c} * FutureSysFieldingProfile_{\mathcal{J},c,t} \tag{6.168}$$

- Constraints (6.169) and (6.170) ensure that the number of future systems flowing in for each component must not exceed the number of non-future systems in service that are being replaced. This fulfills the **Outflow Availability** business rule in relation to future systems.

$$\forall (i,m,c) \in Roles$$

$$\sum_{\substack{\mathcal{J},t: \\ (i,\mathcal{J},m,c)\in FutureTransitions \\ T<t\leq\mathcal{T}}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \leq NumGpInMission_{i,m,c,T} \tag{6.169}$$

$$\forall (i,m,c) \in Roles, 1 < t \leq T$$

$$\sum_{\substack{\mathcal{J}: \\ (i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \leq NumGpInMission_{i,m,c,t-1} \tag{6.170}$$

- 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.

$$\forall \mathcal{J} \in MandatedFutureSys, c$$

$$bFutureSysComponent_{\mathcal{J},c} + bFutureSysDeficit_{\mathcal{J},c} = 1 \tag{6.171}$$

- 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.

$$\forall \mathcal{J}, c$$

$$iFutureSysMandateDeficit_{\mathcal{J},c} = \sum_{t} FutureSysFieldingProfile_{\mathcal{J},c,t} * bFutureSysDeficit_{\mathcal{J},c} \tag{6.172}$$

- Constraints (6.173) and (6.174) ensure that the $bFutureSys_{\mathcal{J}}$ indicator variable equals 1 if and only if future system $\mathcal{J}$ is activated for any component c. These constraints partially fulfill the **Future Minimum Group Transition Density** and **Future Minimum Group Final Density** business rules.

$$\forall \mathcal{J}, c$$

$$bFutureSys_{\mathcal{J}} \geq bFutureSysComponent_{\mathcal{J},c} \tag{6.173}$$

$$\forall \mathcal{J}$$

$$bFutureSys_{\mathcal{J}} \leq \sum_{c} bFutureSysComponent_{\mathcal{J},c} \tag{6.174}$$

- Constraints (6.175) and (6.176) ensure 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 for any component 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.

$$\forall(\mathcal{J},m,\ell) \in FutureUpgDensityFlags$$
$$\sum_{\substack{i,t:\\(i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \geq (bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} + bFutureSys_{\mathcal{J}} - 1) * \ell \tag{6.175}$$

$$\forall\mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J},m,\ell) \in FutureUpgDensityFlags$$
$$\sum_{\substack{i,t:\\(i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \leq \sum_{\substack{\ell:\\ \ell\in UpgDensityLevels_m\\ \ell\neq\max(UpgDensityLevels_m)}} (bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} * \ell) + \sum_{\substack{\ell,c:\\ \ell\in UpgDensityLevels_m\\ \ell=\max(UpgDensityLevels_m)}} \left(\begin{array}{l} bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} * \\ GpPerComponentMission_{c,m} \end{array} \right) \tag{6.176}$$

- Constraint (6.177) ensures that future system $\mathcal{J}$ in mission m over all components can satisfy at most 1 of the future minimum transition density levels. Note that if future system $\mathcal{J}$ never transitions into mission m in any component at any time, then all three $bFutureTransitionedToDensityLevel$ binaries will be 0. Together with (6.175) and (6.176), this fulfills the **Final Minimum Group Transition Density** business rule.

$$\forall\mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J},m,\ell) \in FutureUpgDensityFlags$$
$$\sum_{\ell\in UpgDensityLevels_m} bFutureTransitionedToDensityLevel_{\mathcal{J},m,\ell} \leq 1 \tag{6.177}$$

- Constraints (6.178) and (6.179) ensure that the $bFutureHasFinalDensity_{\mathcal{J},m,\ell}$ indicator variable equals 1 if and only if future system $\mathcal{J}$ in mission m over all components 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.

$$\forall(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$
$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \geq (bFutureHasFinalDensity_{\mathcal{J},m,\ell} + bFutureSys_{\mathcal{J}} - 1) * \ell \tag{6.178}$$

$$\forall\mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$
$$\sum_{\substack{i,t: \\ (i,\mathcal{J},m,c)\in FutureTransitions}} iNumGpReplaced_{i,\mathcal{J},m,c,t} \leq \sum_{\substack{\ell: \\ \ell\in FinalDensityLevels_m \\ \ell\neq\max(FinalDensityLevels_m)}} (bFutureHasFinalDensity_{\mathcal{J},m,\ell} * \ell) + \sum_{\substack{\ell,c: \\ \ell\in FinalDensityLevels_m \\ \ell=\max(FinalDensityLevels_m)}} \left(\begin{array}{l} bFutureHasFinalDensity_{\mathcal{J},m,\ell} \\ *GpPerComponentMission_{c,m} \end{array} \right) \tag{6.179}$$

- Constraint (6.180) ensures that future system $\mathcal{J}$ in mission m over all components can satisfy at most 1 of the final transition density levels. Note that if future system $\mathcal{J}$ never transitions into mission m in any component at any time, then all three of the $bFutureHasFinalDensity$ binaries will be 0. Together with (6.178) and (6.179), this fulfills the **Future Minimum Group Final Density** business rule.

$$\forall\mathcal{J} \text{ where } m = FutureMissionMap_{\mathcal{J}} \text{ and } \exists(\mathcal{J}, m, \ell) \in FutureFinalDensityFlags$$
$$\sum_{\ell\in FinalDensityLevels_m} bFutureHasFinalDensity_{\mathcal{J},m,\ell} \leq 1 \tag{6.180}$$

DISTRIBUTION:

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

1 Frank M. Muldoon
Applied Materials
5225 West Wiley Post Way, Suite 275
Salt Lake City, UT 84116

1	MS 1188	Dean A. Jones, 8830 (electronic copy)
1	MS 1188	Bruce M. Thompson, 8833 (electronic copy)
1	MS 1188	Craig R. Lawton, 8834 (electronic copy)
1	MS 1188	Alan S. Nanco, 8836 (electronic copy)
1	MS 1188	Dennis J. Anderson, 8836 (electronic copy)
1	MS 1188	Lucas A. Waddell, 8833 (electronic copy)
1	MS 1188	Stephen M. Henry, 8833 (electronic copy)
1	MS 1188	Matthew J. Hoffman, 8831 (electronic copy)
1	MS 1188	April M. Zwerneman, 8834 (electronic copy)
1	MS 1397	Peter B. Backlund, 5837 (electronic copy)
1	MS 1188	Darryl J. Melander, 9365 (electronic copy)
1	MS 0899	Technical Library, 9536 (electronic copy)

This page intentionally left blank.


Sandia National Laboratories