2011 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING AND INTEGRATION (SE) MINI-SYMPOSIUM
AUGUST 9-11 DEARBORN, MICHIGAN

Hierarchical Platforms-Based Design of Ground Vehicles

Dr. Sandeep Mehta
CEO
NspiRD
El Segundo, CA

Dr. Stephen Cooper
Systems Analysis
General Dynamics
Sterling Heights, MI

An innovative new approach is presented that addresses the challenges of design in a constantly changing environment. New solutions that satisfy changing requirements are generated by rapidly reconfiguring ongoing projects and effectively reusing trusted designs. Design is essentially a process of generating knowledge about how to build new systems. Reuse is difficult because this knowledge is amorphous and difficult to access. Hierarchical platform-based engineering is used to structure and categorize this knowledge to make it easily accessible. This approach has three essential components: 1) Hierarchical platform-based design method organizes design projects into a structured library; 2) Transformational systems engineering and concurrent risk assessment are used to capture complex interactions between different CPS elements. These captured interactions help assess reusability and reconfigurability of each element; 3) ***A*** *new design flow integrates platform-based design methods into the overall design workflow. Detailed examples are presented to illustrate the use of the new approach.*

INTRODUCTION

Rapidly changing environment and asymmetric threats require increasingly complex ground vehicles (GVs). The need for specializations is driving towards customizations and ever smaller lots. This combination of complexity and customization has clearly demonstrated the limitations of traditional requirements-based development methods. As shown in Figure 1, the development of new DOD systems has experienced rapid exponential cost growth [1]. Even with the increasing costs, many development programs have faced significant delays and cost overruns. A recent government accountability office (GAO) report [2] on 95 weapons systems programs found total cost growth of $295 billion with an average schedule delay of 21 months. Problems with complexity are not just limited to weapons. For example, recent quality problems at Toyota were also shown to be directly related to increased complexity [3].

A new approach is needed that can address challenges of complexity and customization. An example of successful development methods is in electronics systems (e.g. Integrated Circuits or ICs). IC design has managed complexity well and delivered increasingly capable products at continually reducing costs. A key contributor towards the success of ICs has been reuse of designs. The electronics design reuse is supported by platform-based design (PBD) methods employing structured decomposition based on rigorously-enforced architecture.

The next section provides an overview of development of PBD methods. The following section outlines the challenges in implementing electronics PBD in GV design. A new hierarchical platforms-based approach is presented that addresses these challenges to enhance design reuse across complex physical systems such as GVs. Several challenges and benefits of implementing the new PBD methods are discussed. Finally, a detailed example is presented to illustrate the use of PBD methods in development of diverse ground vehicles.

HISTORY OF PLATFORM-BASED DESIGN

Product families or platforms have been used for many years. A key use of platforms has been to support a large variety of product while managing cost and schedules. The emergence of global markets and related competition has compressed product development times and increased the focus on platform-based design [5].

The term platform has been used by car makers (e.g. Chrysler K-car platform of the 1980s) to denote models sharing common features, subsystems or components [9]. Platforms have also been used in aerospace design (such as Boeing 737 family of aircrafts). PC makers have been able to develop their products quickly and efficiently around a standard "platform" that emerged over the years [13]. In the case of PC, the platform has become synonymous with interface and architectural standards. The PC platform

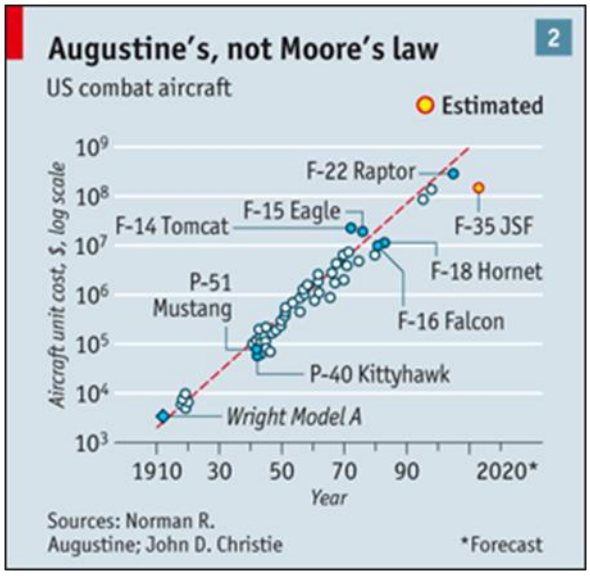

Figure 1: US Combat Aircraft costs [1]

standards have allowed significant reuse and generated an entire ecosystem of developers.

Hence, platforms have been defined as a set of subsystems and interfaces developed to form a common structure from which a stream of derivative products can be efficiently produced [10]. Three types of platforms have been identified: 1) Modular platforms [6] that provide elements that can be combined together to form new instances or designs; 2) Scalable platforms that can be stretched or shrunk to form new designs; and 3) generational platforms that cover different generations of products [11].

Some of the recent innovations in PBD were generated for design of integrated circuits (ICs) or system-on-chip (SOC) [4]. In that context, a platform is a library of components that can be assembled to generate a design at the corresponding level of element [13]. A platform is considered a flexible IC where customization for a particular application is achieved by programming one or more of the components of the chip. Programming may imply metal customization (gate arrays), electrical modification (FPGA personalization), or software to run on a microprocessor or a DSP [9].

Definition of the platform architecture is critical to the success of PBD. The idea is to have a stack of platform layers where the upper layer is an element of the layer below. The lower layer is the set of rules that allow one to classify a set of components. The library is in some sense a parameterization of the space of possible solutions [13]. A platform contains possible solutions to a design problem that share a set of common features. The decisions taken at a higher level define the configuration of the system at a lower level [13].

PBD in electronics is a hierarchical methodology that integrates design reuse into the design flow. Each element of the IC is developed with reuse as one of the objectives. Furthermore, interfaces between different elements are standardized to further facilitate reuse. The overall architecture of the platforms is also developed with reuse in mind. Each element in the platform library is tested and verified for a wide variety of uses to ensure that a new design using the library will also be correct by design.

One of the first examples of the modern use of PBD is in TI OMAP SOC development. [13] ST Microelectronics defined platform-based design as the creation of a stable microprocessor-based architecture that can be rapidly extended, customized for a range of applications, and delivered to customers for quick deployment [7]. ST attributed continued success of its set-top business to the use of platforms for its systems-on-chip applications [14].

The success of PBD in IC development resulted in the desire to use the same approach for other parts of systems that use electronics. To that end, the next implementation of modern PBD was in embedded systems where an IC or a SOC is designed to interface with and monitor/control some physical function (such as cruise control). Embedded systems control the physical processes, usually through feedback loops, where physical processes affect computations and vice-versa.

The next big challenge is the implementation of PBD in cyber-physical systems (CPSs) [8]. CPSs are systems that integrate mechanical, electrical, optical, chemical, and digital/software elements. From a PBD viewpoint, embedded systems are orders of magnitude more complex and more powerful from a computational viewpoint; CPS will probably be yet another order of magnitude more complex and powerful. CPS will allow developing a wide span of applications because of the availability of a new generation of sensors, actuators, and local computing that leverage novel interconnect capabilities and centralized computation. The key challenge is about the interfaces and communication patterns between the computing and the physical systems as well as about the functions that a rich ensemble of heterogeneous entities can implement [10].

Platforms have allowed companies to reach the promise of mass customization – providing products for a large market while customizing them for the desires of every individual customer. Platforms have the additional advantage of maintaining commonality, compatibility and modularity between a wide array of products. Platforms also provide an effective structure to review and manage a spectrum of related products. Platforms promote better learning across products by sharing development resources. Platforms can reduce certification costs of complex products such as ground vehicles, aircrafts, or engines.

In automotive industry, use of platforms can yield a 50% reduction in capital investment and reduce the product lead times by 30% [5]. Automotive manufacturers that used product platforms in the 1990s gained a 5.1% market share while those that did not, lost 2.2%. Volkswagen saved an estimated $1.5B per year in development costs by using platforms.

However, platforms also have some disadvantages: Over dependences of platforms can make all products similar. Too much commonality can dilute brands across the entire platform. A focus on commonality during platform development can increase development costs by 100% to 1,000% [5]. Platforms also involve significant changes to the processes and tools used in design. [14] The next section describes some of the challenges involved in implementing PBD in complex systems such as GVs that are dominated by electrical / mechanical elements.

CHALLENGES IN IMPLEMENTING PBD FOR GV

Advanced PBD tools have been critical to the successful increase in electronics capabilities at reduced costs. Languages such as VHDL model electronics systems and facilitate communication between engineering, manufacturing and quality control. Advanced simulation tools allow electronics designers to simulate most aspects of system performance and eliminate expensive testing.

PBD methods have been extended to embedded systems. However, embedded systems PBD are incapable addressing CPSs such as GVs. Embedded systems PBD appraoches consider physical elements adjunct to electronics / software and only as those providing feedback to software. Furthermore, CPS such as GV have much higher reliability requirements, consequences of failure are much larger and operating environment / parameters are not predicatable [8]. Several challenges prevent simple implementation of electronics-like PBD methods in CPS. A thorough understanding of these impediments is necessary to develop an approach that transcends them:

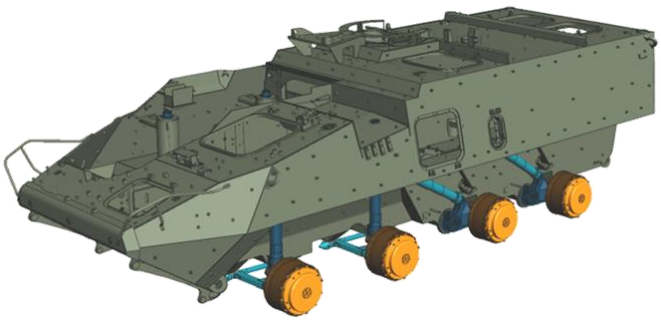

Figure 2: GVs such as GD Stryker are complex CPSs

Flexibility in definition of platform architecture

Disciplined, rigorous and consistent definition of platforms is critical to their reuse. Electronics and software systems are easily divided into components because they are inherently logical. CPSs have much more flexibility around definition of platform layers (subsystems, components or configurations items). This flexibility leads to variability in hierarchical representation of systems across design teams and reduces reusability of platforms.

In many cases, different teams within the same organization may use different architecture to represent the same system. Sometimes different skill-sets within the same development team use different hierarchical representations of the system. For example, system engineers break down an engine into one set of functional elements, designers model the system using manufacturing-related element, life engineers may break it down into different groupings based on material properties and computational fluid dynamics engineers may build yet another set of elements.

This variability means that designs are not easily reusable because each layer means something different in each instantiation of the system. The next section describes a new approach for definition/reuse of platform architecture so they can be reused just like blocks in an ASIC.

Complexity of interactions between elements

Use of a platform in a new system requires the ability to quickly evaluate its capabilities and its fit with other parts of the system. In software development, developers define and control interactions between modules and objects. In electronics, designers can follow standards for interfaces between different elements. These standards define how components interact with each other and make it possible for projects from platform libraries to be mixed and matched.

Interactions in CPS elements are governed by the underlying physics and not easily controlled. While some of the interactions are obvious, many are complex, non-linear and difficult to identify. Years of experience goes into generating a knowledgebase of interactions that need to be addressed in design. Even so, many interactions remain unknown at the time of design and are only discovered during testing or deployment.

Design teams spend significant effort identifying these interactions because they drive system behavior, performance and failure. Once interactions and failure modes are known, numerical models can be developed to analyze the underlying physics and predict system behavior. In fact, these interactions contain the design logic or rationale of CPS design. Some of these interactions are identified through systems engineering and requirements flow-down. While others are only captured informally and embedded deep inside detailed design documents. Yet others are defined in risk assessment or during testing.

The reuse of a project from the platform library will require effective evaluation of its interactions with other elements being selected in the new design. This is a complex problem because the design reuse may involve a new environment which may change the likelihood of the element failing or impact its form/fit/function. Furthermore, new combination of elements in a design may initiate new failure modes because of previously unobserved interactions. A new CPS PBD approach is presented in the following section that addresses the issue of complex interactions.

Diversity of technologies and skill-sets

It is somewhat easy to compare and select different projects from a platform-library in electronics because of similarity of technologies, engineering disciplines and manufacturing methods. However, technologies involved in most CPS are diverse. For example, a ground vehicle may include mechanical, electrical, optical, electronics and chemical (fuel) subsystems amongst others. A wide range of materials is also involved: metals, ceramics, composite materials, semiconductors, etc. The engineering disciplines involved range from mechanical to material scientists to systems. Another example can be a laser weapon system that encompasses diverse technologies such as laser diodes, crystalline materials for gain modules, optics for beam combining, and electronics for control.

A key requirement for reusability is a simple comparison of existing designs from the platform library. Electronics components have the advantage that only a handful of parameters can describe their designs (number of gates, frequency, power, etc.). Similarly, software development can be described using lines of code and input parameters. When technologies are somewhat related, it may be possible to approximately compare metrics across technologies. For example, in systems involving primarily electronic circuits and software (such as embedded systems), metrics of interest are number of gates on ASIC, number of layers on a circuit board and number of lines of code – each of which is similar in character and somewhat easy to compare. [15]

CPSs utilize completely distinct technologies and engineering disciplines, which makes it more difficult to compare designs. For example, in a laser weapon system development, thermal engineers may measure rate of heat dissipation while the optical designers may focus on beam quality. In ground vehicles, material scientists may measure grain structure to reduce flaw sizes while manufacturing engineers may measure ability to machine parts. Goals of different disciplines are often at odds with each other and a system of metrics is needed that can help provide a unifying objective.

The rate of improvement is different between technologies and further reduces effectiveness of metrics. For example, a laser system uses both electronics and crystalline materials. Performance in electronics is increasing much faster than crystalline materials. Advances in each discipline and improvement in discipline-specific tools is exacerbating these gaps. Hence, metrics need to be easily updateable as different parts change at different rates.

In electronic systems, it is somewhat simple to link manufacturing to overall design through process characterization libraries. These libraries have not been developed for electromechanical systems because the diversity of manufacturing processes makes a universal representation difficult. Many weapon systems require very low rate production and use a job-shop manufacturing approach where the same machine is used to manufacture many different parts. This makes characterization of manufacturing processes even more difficult. Furthermore, since desired product characteristics are separated from required manufacturing capabilities through layers of engineering models, it is difficult to use process libraries even if they were available. Hence electronics PBD methods have to be modified to be independent of technologies, disciplines or manufacturing methods. The link between design and manufacturing has to be tracked across the system hierarchy.

HIERARCHICAL PLATFORMS FOR GV DESIGN

The development of cyber-physical systems (CPSs) such as ground vehicles is complex and requires a wide array of subsystems and technologies. In a changing environment, the impact of changed objectives needs to be segregated to each system element. Since new development is expensive, it is essential that organizations quickly identify ongoing design projects that can be reconfigured or completed designs that can be reused to satisfy changed requirements.

Product Platforms	Technology Platforms
Hierarchical structure: : System→Subsystem→Sub-subsystem etc	Non-hierarchical structure
Each platform can represent a hierarchy of sub-platforms	No hierarchy
Each project is initially intended for one product	Each project supports multiple products
Possible to compute financial metrics such as return on investment	Difficult, if not impossible to measure financial returns
Examples: Ground Vehicles →Power Trains→Engines…	Examples: Titanium Casting, Coating, Carbon composite

Figure 3: Product and Technology Platforms

Platform-based design (PBD) is essentially a library of related design projects that can be used to generate a new design. The key to the success of platform-based design is the overall architecture or the taxonomy of the platforms.

The platform architecture essentially defines the potential system configurations. As shown in Figure 3, two types of platforms are utilized to create an innovative new approach for CPS PBD: Product Platforms and Technology Platforms. Product platforms are hierarchical and consist of subsystem, sub-subsystem platforms and so on. Product platforms encompass design projects that are delivered directly to the customers and where it is possible to measure the return on design investment.

Technology platforms are common across products and get embedded into components within product hierarchy platforms. Since technology platforms support multiple platforms, it is not possible to compute financial returns on related projects. Each product hierarchy platform can have sub-platforms (products or technology), while technology platforms remain independent. Technology projects tend to be much more long-term and high risk. This dual platform architecture allows CPS design teams to maintain configuration control on platforms while segregating development into meaningful bins. For example, a product platform for a combat vehicle engine manufacturer could be Piston Engines; subsystem platforms could be Engine Blocks, Ignition Subsystem, Fuel Injection Subsystem, etc.; technology platforms could be Castings, Liners, etc.

As shown in Figure 4, platforms are libraries of design projects at the corresponding level of system configuration hierarchy. The Engine platform could encompass design projects for different types of engines (displacements, number of cylinders, etc.). The block Platform could encompass different types of blocks (cast iron, cast aluminum, etc.). The new approach divides a system design project into several linked subsystem projects (corresponding to available subsystem platforms). Each subsystem project is divided into linked projects for sub-subsystem or component platforms, etc. Continuing the previous example, a product design could be an upcoming twelve-cylinder diesel engine. The engine project would be divided into several subsystem development projects. One of the subsystem projects could be an aluminum block for the diesel engine, and a sub-subsystem project could be a liner for the cylinder. A component project could be ceramic materials for the liner.

Since each system project can only consist of projects corresponding to defined subsystem platforms, the new PBD method brings discipline and rigor to CPS decomposition. If there are no subsystem configurations that can satisfy the design needs, the organization has to consciously define a new subsystem platform that does. For example, if the platform hierarchy contains no cylinder liner element, its creation would require a change in the platform structure and all relevant parties would need to be informed. This rigorous platform structures controls configurations and ensures that all related development is easily accessible

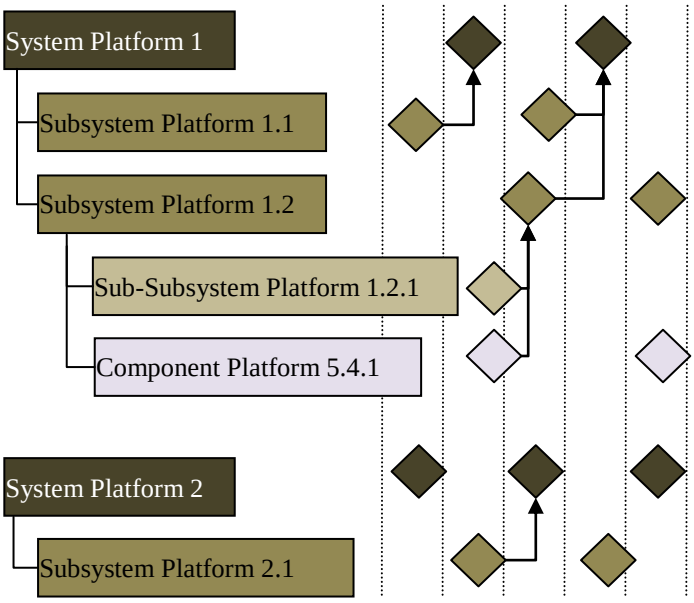

Figure 4: Hierarchical platforms organize design projects

across geographical boundaries. This platform structure enables organizations to leverage synergies between similar design projects and provide consistent oversight across organizations.

The hierarchical structure of platforms and projects allows deep reuse at any level: systems, subsystem, sub-subsystem or technology. Depending on the need, an organization can develop a new system using any combination of projects at different levels of system hierarchy. If environment change leads to different objectives for the system, the design team can either choose to reconfigure the block development project or look at all other blocks (in development or trusted) and evaluate if they can be reused more effectively.

Traditional PBD involves changing all design projects to include reuse. This change automatically incurs extra development costs for projects (up to 1,000%). The new approach automatically characterizes all projects and makes them available for reuse or reconfiguration. This approach avoids unnecessary broadening of design objectives just to satisfy future reuse. Furthermore, the new approach allows use of PBD for all modular, scaled or generational platforms [10]. In fact, the new approach can support all integral or modular products. Within modular products, all types of modularity are supported: component swapping, component sharing, fabricate-to-fit, bus, sectional and mix [10]. Finally, this approach is applicable regardless of technologies, materials, manufacturing processes or engineering disciplines. The next section further enhances the applicability of the new approach by capturing and modeling complex interactions across product hierarchy.

NEW PBD MODELS COMPLEX GV INTERACTIONS

Coupled interactions between materials / components / subsystems lead to uncertainty in system performance. Traditional development addresses this problem by qualifying the system as a whole, with all manufacturing processes, materials and environment, often through build-test-break methodology. This traditional approach does not support platform-based design. Furthermore, this qualification process inhibits the insertion of new technologies, materials and innovations into the system.

The new PBD approach[1] can be successful only if these complex, coupled and often nonlinear interactions between platform library designs can be evaluated. Unfortunately, these interactions are generally only expressed informally in design decisions and not available as formal quantitative measures that can be incorporated into a model. The new PBD approach implements an innovative new approach to capture these interactions

As shown in Figure 5, design is essentially tasks undertaken to ensure the system fulfills its objectives and mitigate risks of not meeting those objectives. Objectives are parsed out to different elements of the system through requirements flow-down. The links between requirements represents one set of system interactions. Risk assessment is utilized to identify and mitigate risks of system failure. The flow of events causing system failures represents the other set of system interactions. The new approach links requirements flow-down to risk assessment and makes the resulting information available for each design project. The framework of objectives-requirements-risks transcends all disciplines involved in design and manufacturing of a CPS. Hence, these captured interactions represent the essential CPS design rationale or the design logic.

Existing systems engineering and design processes limit requirements flow-down to larger subsystems and components. Many of the interactions driving individual part designs are lost in this approach. Transformational systems engineering is used to link requirements across the system hierarchy: from overall system to subsystems to individual components to discipline-specific models. All design team members regardless of discipline or role will participate in transformational systems engineering and link their design objectives to parent requirements.

Traditional design segregates formal risk assessment into a separate process that is only loosely linked to the overall design flow. Most risk mitigation interactions are informal and not captured anywhere. A concurrent risk assessment process is used to formalize and integrate risk mitigation into the overall design flow. The new process extends risk assessment to the entire design team. Frameworks such as FMEA (Failure Mode and Effects Analysis), fault-tree analysis and event sequence diagrams are used to help elicit and capture risk-related inter-element interactions.

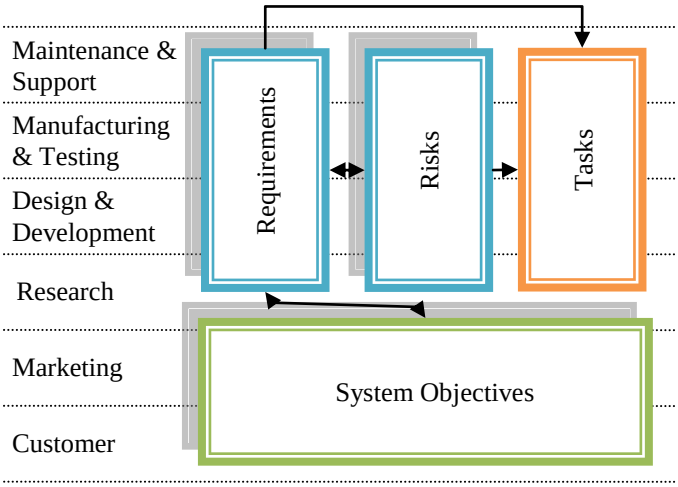

Figure 5: Intuitive capture of interactions

As shown in Figure 5, this new approach to capturing interactions is completely independent of technologies, engineering disciplines, materials or manufacturing processes. This approach also integrates well into the traditional physical development design flow – leveraging existing skills of systems engineering and risk assessment. Furthermore, existing design processes can be modified to quickly evaluate the reuse of an existing design in a new system. In fact, implementation of PBD will require modification of several key processes involved in traditional CPS design flow. Each is discussed in the next section.

PBD IMPACTS MANY GV DESIGN PROCESSES

PBD can significantly reduce costs of developing new systems. However, several researchers have noted that implementation of PBD requires a significant change in processes followed in traditional design organizations [12]. Our new PBD approach minimizes these changes while ensuring that any required change brings improved efficiency and effectiveness.

As described above, configuration management, systems engineering and risk assessment need to be modified to implement PBD. These updated processes will not only support PBD, but also enhance efficiency of GV design processes such as project reviews and program management. The new objectives → requirements → risks → tasks approach provides comprehensive context for project reviews. Project reviews automatically become hierarchical – projects corresponding to each lower level platform feeding into higher levels. Furthermore, project reviewers are able to quickly ascertain requirements and risk driving the design effort and results. All relevant review feedback could be rapidly integrated back into the overall design

[1] Patent Pending

rationale. Since our new approach provides a clear context for every task being undertaken, program managers are able to more effectively decide on which tasks are critical to overall progress. Program managers are able to quickly perform a cost benefit analysis on every action based on the benefit they may have to satisfying overall system objectives or mitigating risks.

Our new PBD will further enhance efficiency by more effectively integrating with ancillary processes such as pricing, project control and skill-set management. The objectives → requirements → risks → tasks structure replaces the traditional WBS (Work Breakdown Structure) used for pricing and controlling project costs. Hence every constituency involved in managing the design project communicates using exactly the same taxonomy. Furthermore, stakeholders are able to tie costs to particular customer requirements or system objectives. Delays or problems are also associated with overall design goals. Our approach integrates and enhances all metrics involved in controlling programs such as CPI or SPI.

The platform-based hierarchy along with the objectives → requirements → risks → tasks structure can enhance skill-set management functions such as performance reviews. Each task has a rationale for why it was undertaken and the performance of each team is more meaningfully captured.

One of the biggest impacts of PBD is on planning and resource allocation for design projects. The platform architecture provides the structure to collect new development plans from across a large organization (such as that involved in GV development). The plans are arranged automatically and are available for communication / sharing. The platform structure also facilitates a new level of collaboration by making new connections between researchers working at different levels of platform hierarchy. For example, an armor development engineer is able to see all new material development occurring across the entire organization regardless of driving system or program. Similarly, material developers are able to understand the driving requirements in new armor development projects and are able to modify their projects accordingly. Platform libraries provide decision makers detailed insights into proposed projects and their ultimate benefits and allow them to make more informed investment decisions.

DESIGN FLOW INTEGRATES NEW PBD

Figure 6 shows an updated design flow that incorporates PBD. Each step of the design flow is explained below.

System Objectives drive PBD

The design of complex CPS such as a GV begins with the identification of overall system objectives. In this step, designers work with customers, marketing personnel or technologists to define the desired system functionality.

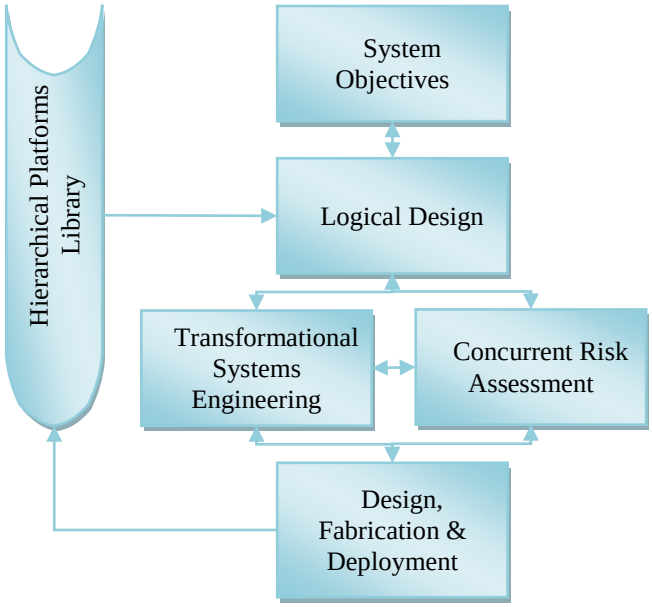

Figure 6: Hierarchical platform-based design flow

Objectives are used to create a logical design of the system. The objectives also drive requirement flow down, risk assessment and detailed design.

Logical design using Platforms

Given the system objectives, a designer must create a logical design of interconnected components, each component implementing a portion of those objectives. Logical design is an iterative process where available platforms are evaluated for suitability of their use in the new system. Platform library is used to build a new logical design. As with electronics design, the use of the platform library drives reuse and reduces development costs.

The hierarchical platforms library also controls configuration of new systems that can be developed based on the platform architecture. In instances when the new element being developed is logically different from all existing abstractions, a formal process exists to create a new platform and integrate it into the overall architecture. The new platform can then be used just like all existing ones.

Transformational Systems Engineering

Requirements flow-down links overall system objectives with subsystem/ component requirements through the logical design. This process is normally part of system engineering. The hierarchical platforms library allows designers to automatically pull together requirement flow-down from previous design instantiations. If the new requirements are satisfied by the existing design, no further work may be necessary. If there are mismatches, designers can either follow the traditional system engineering process and

reallocate requirements or start detailed design processes to modify the design to satisfy requirements.

Concurrent Risk Assessment

Requirements flow-down is a somewhat linear process – starting from the highest level of abstraction (overall system) and moving down to the lowest level of detail (component). Risk assessment identifies and addresses complex multi-level, non-linear and often discontinuous / brittle interactions between CPS elements. Identification of these interactions is important because component manufacturing process change may have impact a disparate subsystem.

Hence, the new design flow integrates risk assessment to help designers understand cross-system interactions. The selection of a design from the library automatically pulls together risk assessments from previous instantiations. If there are mismatches, the designers can either update the design or introduce other measures such as inspections to reduce risk.

Design, Fabrication and Deployment

Normal processes can be used if a new design is needed. The updated design can be integrated into the library.

EXAMPLES ILLUSTRATING THE USE OF NEW PBD

This section illustrates the use of the new PBD methods and the design flow in GV design through an example.

Hierarchical Platforms Architecture and Library

Figure 8 shows an illustrative example of a hierarchical platforms library for ground vehicles. This library can be structured so as to support many different classes of GVs from fighting vehicles to battle tanks.

Platform architecture is hierarchical: starting from the system (ground vehicle) and continuing to higher level of details such as subsystems. Subsystem platforms include all types of subsystems such as ground interface. Each subsystem platform is further divided into sub-subsystem platforms. For example, the ground interface subsystem platform could include steering, suspension, tracks, wheels, skis / skids. Referring to Figure 8, six example subsystem platforms are shown: Hull/Frame, Ground Interface, Powerpack, Drive Train, and System Survivability & Lethality. A full library would have additional subsystem platforms such as power distribution, communications, add on kits and so on.

Subsystem platform boundaries ideally are selected for modularity to facilitate reuse with their required interfaces to other subsystem platforms clearly identified. For instance the powerpack from a tracked vehicle could be reapplied in a similarly sized wheeled vehicle. An armor composition from one vehicle could be reapplied in another. In both cases the associated requirements and manufacturing processes for each could be reused and/or provide a basis for those developed in the new vehicles.

At the higher levels of hierarchy, an item such as an air handling unit will have little or no coupling to subsystem platforms such as Drive Train or Lethality beyond space, weight and power requirements. In many cases the higher level sub-subsystem platform elements can be directly reapplied to a new vehicle as long as the lower level platform interfaces have been accounted for and suitably driven. Of course space and routing considerations will be specific to each vehicle and can provide constraints at every level of hierarchy. However, the interfaces that would be present in any vehicle, i.e., power, signal, cooling, mechanical linkage, etc. are anticipated in the library elements.

System Objectives

For purposes of illustration consider a scenario where a hostile power has developed a super tank with a 150 mm cannon that can defeat all NATO MBT's and is equipped with armor that withstands all NATO tank munitions. In response the DOD issues an RFP for a near term vehicle development to counter this new threat. It will mount an anti-tank derivative of the 155 mm howitzer that has gone into rapid development.

Figure 7 shows an illustrative list of objectives that the vehicle will need to meet. These simplified objectives are used in subsections below to illustrate PBD.

Delivery	Start of Production 2 years, 400 delivered over 1 year.
Vehicle Life	100,000 miles Duty Cycle with 40% on smooth roads, 40% on unimproved roads, 20% cross country
Modularity	Design that can be used as a platform for a family of vehicles. (heavy mortar, NLOS, MGS & Command. MGS to be fielded first.
Reconfigurability	Vehicles to be reconfigurable within 8 hours to satisfy specific objectives (urban armor, mine sweeping, etc.)
Lethality	155 mm anti-tank cannon (MGS variant), two 50 cal RWS all variants.
Survivability	Mine Blast – STANAG 4B, 150 mm High Explosive (HE) and Sabot (KE) on frontal armor, RPG on side and rear, 30 mm gun and Hellfire class airborne missiles on top. Redundancy in power and mobility systems.
Range	300 miles unrefueled, 200 hrs. mean time between failures (mtbf)
Speed	At least 50 mph sustained on a smooth level road

Submerged Fording	20’ depth sustained with snorkel, 30’ depth for 500’
Grade	50% grade at 3 mph
„	„

Figure 7: Illustrative GV Objectives

Logical Design using hierarchical platforms

For a given set of objectives, the organization uses the platform library to develop a new system –selecting the elements that constitute the new system. Using the platform library approach it would be possible to instantiate an entire new vehicle by selecting a compatible set of subsystem choices from the platform library, i.e., diesel engine or fuel cell for primary power source, wheels or track, active or

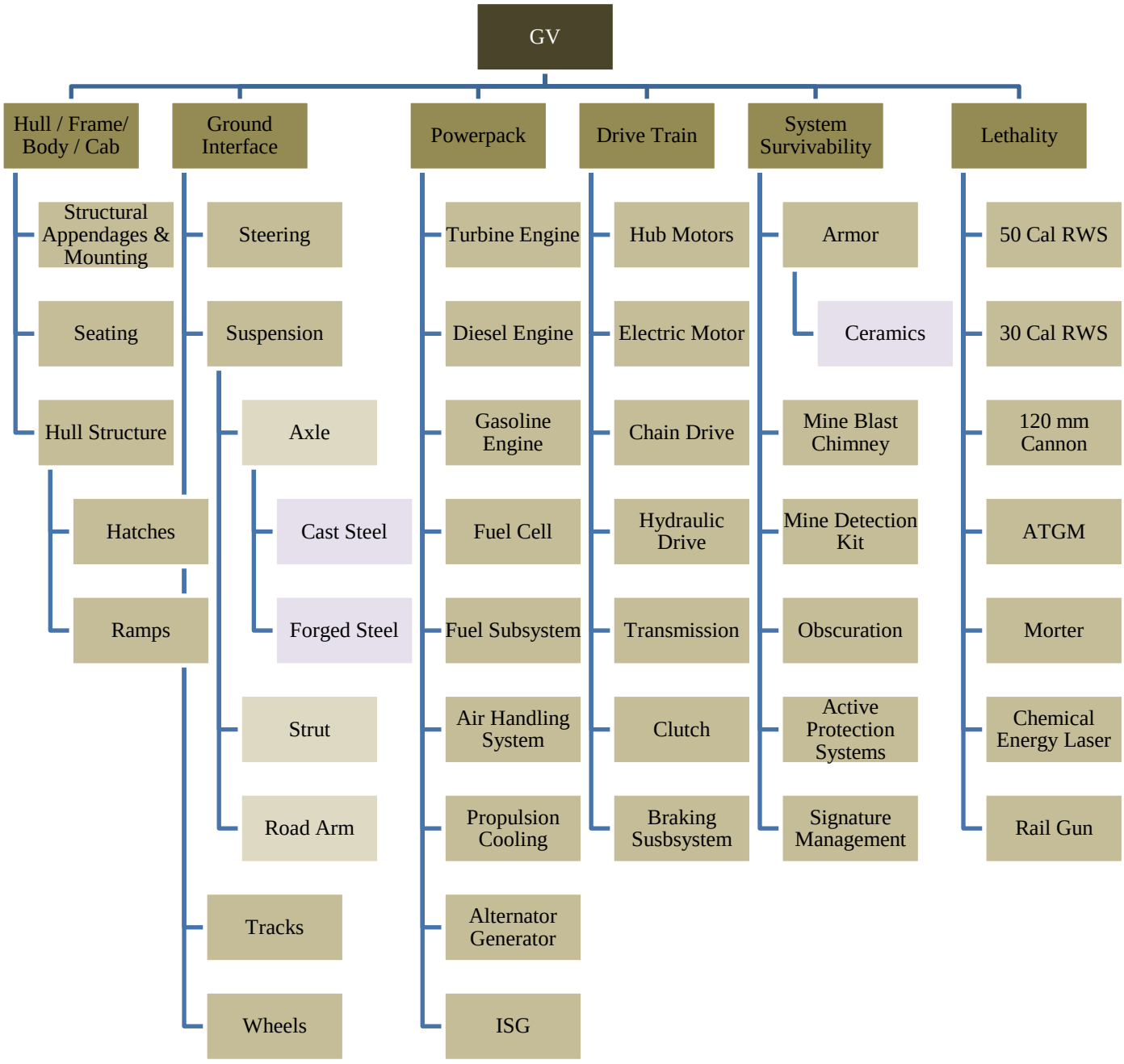

Figure 8: Illustrative Hierarchical Platforms Library for Ground Vehicles

passive suspension, titanium or composite armor, transmission or hub motors and so on. These choices would be driven by the system level objectives for the vehicle being designed. This of course can be done now to an extent in vehicle modeling in a number of vehicle modeling packages such as GT-suite. However, no single modeling tool can span and model all the various domains of a vehicle. The distinction for the hierarchical platforms-based approach is that what is provided from the library are not subsystem models but rather the requirements, risk analyses and related legacy projects that are associated with each subsystem and sub-subsystem platform choice. From these would come the information needed for models, project planning and risk assessment. Additionally, the platform library could be used to create new derivatives from existing vehicle platforms where existing subsystems could be upgraded or modified and/or new capabilities added such as an additional lethality or communication subsystem.

In this example the survivability objectives drive heavier armor which combined with the requirements of the 155 cannon take the vehicle into a new size and weight class. The likely weight of the vehicle will preclude operation on automotive bridges which drives the submerged fording requirement. The aggressive delivery date (2 yrs) precludes the development of advanced/custom tracks/wheels, engines and drive trains, so existing subsystems will have to be reused as is or rescaled within their technology and manufacturability limits wherever possible. This dictates the reuse of existing designs from a hierarchical library to rapidly design a manufacturable and fieldable vehicle.

After surveying existing heavy and medium combat vehicles and the associated subsystem platforms in the platform library, the following configuration based primarily on existing vehicle subsystem platforms in the platform library is selected as a starting point. Referring to Figure 9 it features the following subsystem platform elements:

1. Dual Engine + ISG's (Integrated Starter/Generator) for the powerpack.
2. Outboard tracks with passive torsion bar suspension and inboard strut mounted duals in wheel wells with semi-active (variable damping) suspension.
3. Electric hub motor drives in half of the road stations.
4. Collapsible Snorkel and/or external "drop" chemical oxygen generators with engine exhaust re-breathing to provide gas volume with the oxygen generation.

This configuration provides for redundancy in the power system if one engine is disabled and redundancy in the drive system if a track and/or road-stations are disabled. Once the system has been configured, designers can either reuse designs from the past or use new development from outside or within the military vehicle domain. With the initial configuration proposed, specific subsystem and sub-

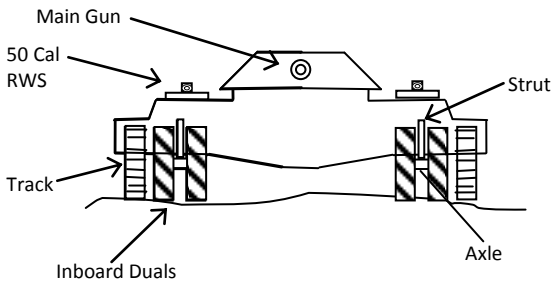

Figure 9: New Vehicle Configuration

subsystem platforms from the library can be evaluated for employment in the new system.

For instance, the Abrams track system combined with a Stryker class wheel system would provide a combined ground interface system that could support the expected weight range of the vehicle. Dual wheels are not a Stryker option but Stryker wheel struts could be combined with dual wheel mounts from dual wheel vehicle subsystems in the platform library. Diesel engines from Abram's weight class vehicles could be reused. The required 900 kW ISG's (Integrated Starter/Generator) would likely be outside the rating of existing ISG's and generators in the library. This could result in the need for a vendor to scale up an existing design or require incorporation of a subsystem from outside the military vehicle domain, i.e., existing locomotive or heavy off road equipment generators / ISG's. An AEV class snorkel from the library would need to be rescaled for the proposed vehicle. The chemical oxygen generators would have to be adapted from existing Air Force prototypes.

Transformational System Engineering

The associated requirements flow-down from each selected subsystem and sub-subsystem platform along with the defined interfaces would provide a basis to create a total integrated requirements flow-down for the new vehicle. Even in the cases such as a chemical oxygen generator where the requirements flow-down would not exist in the library, the defined interfaces to it such as required oxygen and gas volumes for the selected engine would be known. Associated legacy projects "pulled" along with the selected subsystem platforms would provide a basis to develop a schedule and identify the longest lead time components. Tasks from the legacy projects as well as the identified interfaces within and between the subsystem platforms would point to key analyses and tests that would have to be performed to support the integration of the new vehicle.

For instance, a vulnerability analysis could show a problem at the wheel wells where mine blast forces could be focused and trapped within the wheel wells resulting in dangerous floor deformation and vehicle accelerations. To mitigate this effect vertical mine blast chimneys are added

above each wheel well which still allows for useable space around the chimneys. Mine blast chimneys may or may not exist in the survivability subsystem platform and would have to be added if not.

Targeting, mounting, aiming and gun stabilization subsystems from the platform library would exist only for cannons up to 120 mm in caliber. The greater range, recoil and inertia of the 155 cannon would require modification of all these subsystems in the platform library. However, the requirements flow-downs for the existing 120 mm would identify the key issues to address and point to which would be affected by size scaling and which would not.

Concurrent Risk Assessment

In addition to "pulling" requirements flow-downs and legacy projects that are associated with each subsystem and sub-subsystem platform that is selected from the hierarchical library, the associated risk assessments would be pulled as well. For instance, the HV (High Voltage) safety risks that would come with the use of the HV systems associated with the ISG and hub drive subsystems would be essentially the same for the existing ISG's and the proposed 800 kW ISG's. Engine failure modes arising from degradation or blockage in the air handlers would translate directly into the new system. Exhaust re-breathing to support submerged fording would introduce new elements that would have to be integrated with the risk and requirement flow-downs. Once developed this would of course be integrated into the library for potential reuse on other vehicles.

Rescaling of gun and ISG systems would add additional risk items, particularly in meeting the near term delivery objectives. However, the existing risk analysis for the 120 mm and ISG legacy subsystems would provide a good starting point for this analysis and maximize the probability of program success given that these subsystems are necessary to a fieldable design that can meet the customer's high level objectives.

SUMMARY AND BENEFITS

A new approach is presented that brings electronics-like reuse and reconfigurability to GV design. The new process becomes the foundation for a pervasive implementation of platform-based design. The implementation of this hierarchical PBD builds a library of design projects that captures and communicates the design rationale and knowledge across design teams.

This knowledge transfer has an even greater benefit to complex weapon systems design with long product development cycles. Many engineers transition through the same development role in long projects. Each engineer is forced to spend significant amount of time attempting to learn from previous engineers – often unsuccessfully. Knowledge transfer is also critical to successfully transitioning external innovations into delivered products. Finally, the new approach drives collaboration and builds design communities through effective communications.

AUTHOR CONTRIBUTIONS

Framework and methodology development: SM. Ground Vehicle Example: SC and SM.

REFERENCES

[1] Defense spending in a time of austerity, The Economist, August 26, 2010.

[2] Government Accountability Office (GAO), Assessment of major weapons programs (GAO-08- 467SP)

[3] MacDuffie, J. and Fujimoto, T., Under the Hood of Toyota's Recall: 'A Tremendous Expansion of Complexity, Knowledge@Wharton, April 04, 2010.

[4] Chang, H., Cooke, L., Hunt, M., Grant, M., McNelly, A., and Todd, L., Surviving the SOC revolution: A guide to platform-based design, 1999.

[5] Simpson, T., Siddique, Z., and Jiao J., Product Platform and Product Family Design, Springer, 2006.

[6] Gonzales-Zugasti, J., and Otto, K., Modular Platform-based Product Family Design, Proceedings of DETC'00, DETC2000/DAC-14238.

[7] Sangiovanni-Vincentelli, A., Defining Platform-based Design, EE Times, February, 2002.

[8] Lee, E., Cyber Physical Systems design challenges, Technical Report No. UCB/EECS-2008-8, http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html, January 23, 2008

[9] Sangiovanni-Vincetelli, A., Quo Vadis, SLD? Reasoning About the Trends and Challenges of System Level Design, Proceedings of the IEEE, Vol. 95, No. 3, March 2007.

[10] Meyer, M., & Lehnerd, A. P. The power of product platform– building value and cost leadership. New York: Free Press, 1997.

[11] Jiao, J., Simpson, T., and Siddique, Z., Product family design and platform-based product development: a state of the art review, Journal of Intelligent Manufacturing, 18:5-29, 2007.

[12] Sangiovanni-Vincentelli, A., Is a Unified Methodology for System-Level Design Possible?, IEEE Design & Test of Computers, July/August 2008, pp. 346-357

[13] P. Cumming, ''The TI OMAP platform approach to SOCs,[in Surviving the SOC Revolution: A Guide to Platform-Based Design, H. Chang et al., Eds. Boston, MA: Kluwer, 1999, ch. 5, pp. 1–29.

[14] ST steps up set-top box strategy, EE Times, May 2004.

[15] Daniel Gajaski and Frank Vahid, Specification and Design of Embedded Hardware-Software Systems, IEEE Design & Test of Computers, pages 53-67, Spring 1995.