ADAPT DESIGN: A METHODOLOGY FOR ENABLING MODULAR DESIGN FOR MISSION SPECIFIC SUAS

Zachary C. Fisher  
David Locascio  
K. Daniel Cooksey  
Dimitri N. Mavris  
Georgia Institute of Technology  
Atlanta, GA, USA

Eric Spero  
US Army Research Laboratory  
Aberdeen, MD, USA

ABSTRACT

Recent advances in small unmanned aircraft systems (SUAS) have greatly broadened the scope of their potential applications. However, traditional design processes applied to SUAS produce a single design for a single set of requirements. Off-design mission performance is often greatly degraded due to the vehicle’s small scale. This paper considers a different approach to SUAS design aimed at addressing this issue. In this approach, a hybrid modular and scalable product family is coupled with linked engineering analyses in order to automatically formulate a design given a set of mission requirements. This allows multiple SUAS designs to be rapidly synthesized from multiple sets of design requirements using a common set of components. Designs are then rapidly generated and manufactured “on-demand” using automated manufacturing techniques in order to address unforeseen mission needs.

The design approach, named “Aggregate Derivative Approach to Product Design” (ADAPt Design), consists of four actions: (1) requirements analysis, (2) architecture selection, (3) interface design, and (4) concept refinement and design. The outcomes of the method are a family of designs which are highly compatible with design automation, and a toolset that automatically translates changes in requirements to changes in detailed 3-D models. Results of the application of this approach are presented via the design of several SUAS. The capability of the design paradigm is assessed through a comparison of design requirements to the measured performance of the designed vehicle, and conclusions are drawn about the approach’s applicability and scalability.

INTRODUCTION

Presently, U.S. Army UAS are primarily used to support tactical operations through collection of intelligence, surveillance, and reconnaissance (ISR) information. Ideally, troops in the field would employ UAS on-demand to acquire Actionable Intelligence in real-time.

An assessment of U.S. military operations in the suburbs of Baghdad, Iraq conducted by the RAND Corporation for the U.S. Army concluded that modern combat operations increasingly require decentralized decision making. It states “The enemy is fleeting, which means that decentralized decision making is required. Units at the brigade level and below must therefore have access to the information and other capabilities required to support the rapid decisions necessary to deal with a highly mobile enemy … and to enable effective, independent action [1].”

The U.S. Army unmanned aircraft systems roadmap for 2010-2035 supports this conclusion. Furthermore, it recommends that UAS be used to enable decentralized decision making. It states “UAS require and enable accelerated multi-echelon, decentralized decision-making, and execution, significantly changing the tempo and dynamics of operations. Lower echelon leadership must be empowered with authority and bandwidth to employ UAS as their changing situation dictates, operating at a tempo that is faster than higher echelon leadership can affect. [2]”

The U.S. Army Training and Doctrine Command (TRADOC) recognizes that the modern battlespace is a rapidly evolving environment that demands responsiveness from Soldiers and their equipment to maintain dominance [3]. SUAS provide a means to develop situational understanding in support of decentralized decision making during future expeditionary operations envisaged by TRADOC.

Accordingly, SUAS have increasingly been used to provide battlefield situational awareness. SUAS can perform functions such as intelligence, surveillance, and reconnaissance (ISR), security, manned-unmanned teaming (MUM-T), communications relay [4], finding IEDs, identifying enemy combatants [5], and performing advance scouting all with greatly reduced risk to the soldier [4]. The vehicles currently in use by the U.S. Army can be broken into three categories – division level and above, brigade level, and battalion level and below [4]. Oftentimes, it is difficult for personnel at the battalion level and below to procure and use SUAS assets due to the limited quantity of vehicles available to the Army. Many of these vehicles are also limited in the missions that they can perform due to having been designed around a specific set of requirements.
Equipping a Soldier with a SUAS can take one of three approaches:

1) **Multi-mission asset**: One SUAS asset covers all mission needs while sacrificing performance across all missions.

2) **Set of optimized assets**: A set of SUAS assets designed for a subset of specific missions are deployed; troops may need to carry a large number of assets to cover all possible missions.

3) **Asset on-demand**: One-off asset is specifically tailored to and custom manufactured for the mission it will perform.

These approaches are depicted in Figure 1 for three notional performance metrics. The square represents a multi-mission asset designed to operate in the center of the capability space. The black dots show a set of optimized assets occupying discrete points in a slightly wider space. The gray dots show designs that are generated on-demand and tailored to the need at hand. Figure 1 illustrates that an on-demand approach captures the best of the multi-mission and optimized assets approaches: it can cover a diverse range of mission needs without imposing a logistical burden on the Soldier of having to carry a portfolio of assets.

**FIGURE 1: THREE APPROACHES TO SUAS DEVELOPMENT ADDRESSING DIVERSE MISSION NEEDS**

**BACKGROUND AND LITERATURE REVIEW**

The design methodology presented in this paper couples concepts from product family design and reconfigurable system design with recent developments in the fields of automated manufacturing and micro-autonomous systems. A brief overview of these topics and relevant research efforts are described in this section to give context to, and establish a consistent lexicon for the work presented in this paper.

**Automated Manufacturing**

An on-demand approach requires decentralized decision making power and access to automated manufacturing capabilities, as well as supporting doctrine and processes. Current acquisition and requisition methods are incompatible with this vision. Materiel procurement cycles are generally long, requiring an identification of the need and establishment of formal requirements using the Joint Capabilities Integration and Development System (JCIDS), and translation of those requirements into materiel solutions using the process outlined in the Defense Acquisition System. Requisitioning supplies is less arduous, but still requires a formal approval process. New manufacturing techniques such as 3-D printing, consumer-focused computer numerical control (CNC) milling and laser cutting enable rapid manufacturing of one-off systems and parts. These techniques offer the potential for invention, innovation, modification, and manufacture to be forward deployed at the point of need and are an enabler for the on-demand vision of allowing Soldiers to create materiel prototype solutions to unforeseen or unanticipated problems. These manufacturing techniques are already accessible in some capacity as part of the U.S. Army Rapid Equipping Force (REF) Expeditionary Labs (Ex Labs), a team of trained personnel equipped with mobile manufacturing equipment [6].

**Micro-Autonomous Systems Research (MASR)**

Georgia Institute of Technology has collaborated with the U.S. Army Research Laboratory (ARL) in researching capabilities for assessing the operational utility of small autonomous systems assisting at the squad level. Improved systems engineering processes for these systems is the primary focus of the research undertaken in the multi-year MASR effort.

Previous work has explored a multidisciplinary framework built on the simultaneous application of decomposition and re-composition approaches, and was implemented to provide a structured, traceable method for evaluating mission effectiveness of systems of microsystems [7]. This culminated with an Interactive Reconfigurable Matrix of Alternatives, a tool for comprehending the large concept solution space. Fundamental mission requirements included endurance, adaptability, path planning, and communications [8].

Ref. [9], entitled “An Automated Approach to the Design of Small Aerial Systems Using Rapid Manufacturing”, explores development of the systems engineering processes necessary for the development and test of an autonomous system for use within a building’s interior. The work presented in this paper is a direct continuation of the developments in Ref. [9], with a focus on outdoor aerial operations.

**Product Family and Product Platform Design**

A *product family* is a group of similar products derived from a common platform. Individual products belonging to a family are called *variants*, and each variant has a set of distinguishing features which allow it to meet different requirements than other variants [10],[11]. The advantages of using product families to derive new designs stem from the reuse of major design elements. A generic development process for product families is presented by Jiao et al. [12]. This culmination with an Interactive Reconfigurable Matrix of Alternatives, a tool for comprehending the large concept solution space. Fundamental mission requirements included endurance, adaptability, path planning, and communications [8].

Ref. [9], entitled “An Automated Approach to the Design of Small Aerial Systems Using Rapid Manufacturing”, explores development of the systems engineering processes necessary for the development and test of an autonomous system for use within a building’s interior. The work presented in this paper is a direct continuation of the developments in Ref. [9], with a focus on outdoor aerial operations.
the groupings of "components, technologies, subsystems, processes, and interfaces" that form the basis from which variants in a family are derived [13]. Two types of platforms have been identified: scalable platforms where variants are produced by varying scalable design variables, and modular platforms where variants are produced through the exchange of different modules. A characteristic of modular platforms is a one-to-one mapping between functional element and physical components [14]. Variety versus commonality refers to the tradeoff between retaining common features between variants and ability of the product family to meet a wide range of customer needs. Finally, design optimization refers to the set of techniques used to determine values of the design variables which result in a design that best meets objectives established from the customer needs [12].

An important semantic distinction is the one between product architecture and product configuration. Product architecture refers to the arrangement of functional elements into physical units and the interaction between these units [15]. Product configuration refers to the spatial layout of physical components, features, and modules. In the context of a product family, configuration defines the allocation of these elements between product variants [12].

This work borrows concepts from product family design to enable design automation of SUAS. Deriving variants from a fixed, common product platform separates configuration development, which is difficult to automate, from preliminary and detailed design activities which are more readily automated. This directly leads to the possibility of in-situ SUAS development where vehicle designs are tailored to a wide range of requirements.

Reconfigurable System Design

Reconfigurable systems are systems that can reversibly take on distinct configurations through alteration of form or function in order to achieve differing levels of system performance [16],[17]. Often, reconfigurability is employed to permit systems to operate closer to their optimal performance under a wide range of operating conditions by trading between competing performance metrics [16]. The topic of reconfigurable systems was first introduced as a topic of product design research by Olewnik, Brauen, Ferguson, and Lewis, who describe methods to characterize such systems and assess the flexibility they permit during a design process [17]. Literman, Cormier, and Lewis further present a framework to fully characterize reconfigurable design concepts, which require additional information over their static counterparts. Such a characterization framework is needed to compare between concept alternatives [18]. Of particular relevance to this paper are the developments of Patterson, Pate, and German, who consider a UAS which is built from modular airframe components that can be interchanged between flights. The authors demonstrate several approaches to assess the flexibility in UAS performance attained through reconfiguration of the vehicle [19].

The on-demand design philosophy described in this paper exists at the junction of product family design and reconfigurable system design. More specifically, the design philosophy achieves adaptability not by physical reconfiguration of an individual product, but rather through on-the-fly reconfiguration of the product’s design. This notion results in a set of designs that in many ways resembles both a product family and a reconfigurable product.

RESEARCH OBJECTIVES

The on-demand approach is succinctly explained via an analogy to Lego® depicted in Figure 2. Lego® bricks contain a number of modular parts that can be constructed into different models depending on what outcome is desired. Instructions are provided to help the user build different systems out of the same set of components. In the context of this work, a small set of off-the-shelf parts which cannot currently be manufactured on site, such as motors, propellers, and control electronics, will be provided ahead of time to a supply facility at a forward operating base. These off-the-shelf parts will then be combined with parts manufactured on-site to create the needed system.

Figure 3 depicts a potential concept of operations for implementing the on-demand approach to an immediate Soldier need. In this case, the underside of a bridge needs to be inspected using a hover-capable system. The patrol relays their mission needs to a supply facility (e.g., forward operating base or REF Ex Lab), where an integrated engineering workflow is used to design a SUAS tailored to the immediate mission need. Within hours, the solution system is delivered to the Soldiers who use it to inspect the bridge. Alternatively, prior to conducting a patrol, the integrated engineering workflow can be used to create alternate SUAS as individual units of issue. Soldiers would then requisition and pick up the systems from the supply facility prior to departing on the patrol.

Several challenges must be addressed before an on-demand approach can be realized. First, technicians must be trained to use the integrated engineering workflow as well as the available automated manufacturing equipment. Manufacturing technicians are envisioned to be involved in this process primarily because the Soldier’s focus will always be on the mission, but also because the process aims to eliminate the need for a high level of engineering background.

Another challenge is that the design process must be developed in such a way that the vehicles perform as they are intended to. The
implication is that each SUAS designed and developed using the integrated engineering workflow will not be tested before operational use. The intent is to be able to move directly from design to manufacture and then to deployment. This challenge is perhaps best illustrated by the systems engineering “vee” model in Figure 4 which captures the systems engineering actions of a general development cycle. The left leg translates customer requirements into system requirements, and then decomposes the design into increasingly specific subsystems. At each step, a plan is established to verify and validate the resulting subsystems by following the right side of the “vee”. Traditional design processes rely on having an assembled product to conduct verification and validation. This is not the case in an on-demand approach, where the entire re-composition (e.g., verification and validation) leg of the “vee” will have to be collapsed into the decomposition leg. Our approach for developing trusted on-demand systems relies on pre-validating platform architectures, configurations, components, and subsystems, and leverages computer based modeling tools and automated manufacturing techniques.

This paper presents a method to define and architect a set of product platforms that are highly compatible with design automation. In this context, the set of platforms is best described as a product family, except that the product variants exist not as discrete entities but as potential designs that vary continuously over a fixed design space. This is a departure from the traditional concept of a product family, except that the product variants exist not as discrete entities but as potential designs that vary continuously over a fixed design space. In this context, the set of platforms is best described as a product family, except that the product variants exist not as discrete entities but as potential designs that vary continuously over a fixed design space.

The method developed in this work has been named “Aggregate Derivative Approach to Product Design” (ADAPT Design). New designs are derived from aggregations of pre-determined components and design elements. The platforms are a hybrid modular and scalable architecture. Some components can be swapped one-for-one to form a new variant, while others have features that vary continuously. This notion is illustrated in Figure 5.

At its core, ADAPT Design uses rigorous systems engineering techniques to form an executable link between input requirements and an output design. “Executable” is not intended to take on an abstract meaning but instead indicates that the aforementioned link is documented in executable code. The code takes requirements as inputs and uses them to directly drive computer aided engineering and design (CAE/CAD) tools which output detailed models and manufacturing files.

![ADAPT Design Concept Illustration](image)

ADAPT DESIGN METHOD

ADAPT Design is presented in this paper as a linear process divided into four actions: (1) requirements analysis, (2) architecture selection, (3) interface design, and (4) concept refinement and design. It is important to note that in reality, the methodology is like all design processes in that implementation is iterative in nature and occurs over time.

1) Requirements Analysis

ADAPT Design begins by determining and documenting the range of needs to be addressed by the product family. By the conclusion of requirements analysis, the following should be identified:

1) Broad definitions of objectives to be fulfilled by the product family in terms of capabilities. Specifically, this means articulating clearly whose needs will be addressed along with a statement pertaining to how they will be addressed.

2) Key stakeholders in the product family’s use and the requirements and constraints they impose on its development. These requirements and constraints will be used to both bound and validate the product family architecture.

3) Engineering metrics against which derived designs will be measured and compared. These are typically quantifiable characteristics of each design and may include performance metrics, physical dimensions, and required manufacturing time.

2) Architecture Selection

The goal of architecture selection is to identify and define the product platform(s) that will comprise the product family by using the objectives, capabilities, requirements, constraints, and metrics established during requirements analysis. All variant designs will be derived from one of the platforms, and so at least one platform should be identified to cover each of the established capabilities. The number of potential platforms is generally exceedingly large, and so a systematic approach to identifying the most promising platforms is needed. Three approaches are (1) a functional decomposition, (2) a historical search, and (3) requirements space coverage.

In the functional decomposition approach, all of the functions required to achieve the specified capabilities are listed. Components are then matched to each function to establish a list of components that
fulfill all functions. This list is then used to build platforms via a morphological matrix.

A historical search can be used to identify classes of products that have previously been used to achieve the capabilities of interest. The result of this exercise is a list of potential platforms and an understanding of the gaps in capabilities which remain.

The requirements space coverage approach focuses on the capabilities gaps. Here, individual platform concepts are conceived in an attempt to sufficiently eliminate coverage gaps. Each concept is analyzed to understand the capability gap it fills and which capabilities remain unaddressed.

A concurrent application of each of these three approaches is recommended to identify the set of platforms that best meets the established capabilities. Consideration must be given to the tradeoff between variety versus commonality. Increasing the number of platforms covers more requirements at the cost of reduced platform commonality. This in turn increases design overhead and logistics related to procuring and holding parts in inventory.

The next major step in architecture selection is a functional decomposition of the selected platforms, and a subsequent allocation of subsystems and individual components to each function. The resulting list of components and subsystems is then inspected to identify which components and subsystems are common across platforms and which are platform-specific.

Components are further classified as being "modular" or "scalable." A modular component indicates that generating variant designs is achieved by swapping discrete alternatives of that type of component. Modular components will be supplied to the user beforehand and will be used with little or no modification. Scalable components can scale via a limited set of continuous design parameters. Examples include part dimensions, instances of geometric patterns, and locations of features. This classification is not exclusive; components can have nested modular and scalable elements. For example, a beam feature is fabricated by cutting a length of an aluminum tube. Two alternative tubes are supplied: a circular cross section and a square cross section. In this case, the beam’s cross section is modular while its length is scalable.

To document the design decisions made to this point, a formal organization of components should now be built in the form of a component library. The component library enumerates key information related to each component or subsystem. This includes its functional role, its classification as modular or scalable, its scalable variables if applicable, any interfaces, and all engineering data that is pertinent to its use in a design or modeling its performance. Engineering data is subsequently referred to as “attributes.” The component library will serve as the primary source of information regarding available component alternatives as required by automated engineering analyses.

Architecture selection concludes with a step akin to traditional conceptual design. If not already completed as a byproduct of platform identification, a preliminary definition of each platform’s layout should now be established. The result is a set of configuration layouts for each platform that show rough component layouts and bounds on the interfaces between subsystems. Additionally, the scale of the modular components should be identified. For example, a rough SUAS sizing exercise would indicate the range of electric motor sizes that will be of interest.

Of critical importance is that this step abandons the traditional notion that “major design changes are frozen at the end of conceptual design.” The intent of architecture selection is to identify platforms that are highly adaptable and so the conventional notion is counterproductive. Instead, the configuration layouts should indicate which components and requirements will drive interfaces, the magnitude of variation expected for the interfaces, and a rough idea of the location of components, subsystems, and interfaces.

It is recommended that the configuration layouts are documented in the form of “model skeletons,” which are 3-D representations of key geometric planes, points and shapes located in space. Implemented in a CAD environment, the model skeleton approach has been branded “top down design” by some in the community. A model skeleton and the quadrotor SUAS it represents are depicted in Figure 6.

Model skeletons serve two purposes. The first is to aid the designer in capturing the configuration layout of the platform. These models will be heavily refined as development progresses, but explicitly stating key divisions helps to organize the process. The second is that it helps to remove iterative update cycles during 3-D modeling parts of the design process. During the detailed design stages, individual parts will reference the model skeletons for shared geometric features. Automated operations working on a detailed 3-D model can be very slow; operating on a model skeleton is much faster. Furthermore, resultant interface geometry is more likely to be consistent across parts. The skeleton model generated at this point does not have to contain all of the key points or elements that are likely to appear in the final design. Of more importance is capturing the elements that will be tracked as the underlying structure of the platform and how those elements relate to each other.

3) Interface Design

It was previously mentioned that freezing conceptual design changes is delayed relative to traditional design processes. While the concept is intended to remain flexible, the interfaces between flexible components must be clearly described. Interfaces can be geometric, electrical, or logical as in the case of digital communications. A novel characteristic of ADAPt Design and one that distinguishes it from more traditional design processes is an early emphasis on interfaces; locking down interface definitions provides a standardized mechanism to which new candidate variant designs will conform.

Modular components by definition are not modifiable. Therefore, the interfaces of these components are prescribed by the interface standards attached to those components and any derived variant design must adhere to those interface standards. A first step in defining interfaces is therefore identifying the interface standards of the modular components and capturing them in both the component library and model skeletons.

After interfaces of modular components have been documented, control over the remaining interfaces lies in the hands of the designer. The designer should therefore leverage the previously developed configuration layouts to define custom interface standards. These custom interface standards will be applied to all variant designs, and have additional degrees of freedom over their modular component-derived counterparts.

At the conclusion of interface design, all new geometric information generated should be captured in the model skeletons.
Locations of interfaces and parametric interface geometry are design features shared between components. As such, they are best stored in the model skeleton. Updating the model skeletons with points, planes, and representative interface geometry serves to document the interface design work in a form usable by automated design tools.

### 4) Concept Refinement and Design

Concept refinement and design includes many of the design activities normally associated with traditional preliminary and detailed design. In the context of ADAPt Design, elements of these activities differ in three key ways.

First, design activities are not performed manually but rather are encoded into a set of tools and then linked together. The result is a chain of linked engineering analysis tools, models, and automated decision making capabilities. The combination of these linked tools with the previously established skeleton models is the overall enabling mechanism that takes customer needs in the form of capability requirements, automatically converges to a design variant solution, and outputs a detailed set of models and manufacturing files.

The second key difference is that in traditional processes, the bills of materials and manufacturing techniques are finalized post-design, for the purposes of minimizing manufacturing risk and cost, while maximizing manufacturing throughput. ADAPt Design is meant to enable on-demand design based on automated manufacturing techniques using a common set of off-the-shelf components. As a result, all derived design variants inherently conform to pre-established logistics and manufacturing constraints.

The third distinguishing element is the importance of reducing error in all models and analyses. As described previously, the user expects to be able to assemble the design and use it immediately to meet his or her needs. Essentially, the burden of product verification and validation is transferred from the assembled subsystem or product to the models and analyses used in designing the product. Care must be taken to ensure that modeling error is acceptably low so that predictions closely match the observed behavior of the assembled product.

A primary task in concept refinement and design is to refine and supplement the constraints identified during requirements analysis. The goal is to define a complete set of constraints that the design tools will need to enforce. Constraints can be from different categories such as design, manufacturing, assembly, logistics, or regulatory, and should be quantified where possible. Early identification of constraints aids in architecting the design tools in a way that facilitates automation.

The bulk of the design automation is enabled by executable model-based design and development techniques. These techniques and the architecture of the links between them are problem specific. However, it may be beneficial to divide them into two categories: conceptual/preliminary models responsible for determining driving design parameters which control the model skeleton, and detailed design modeling which controls the lower level geometry and brings the design to a point where it can be manufactured.

Conceptual and preliminary modeling tools are responsible for determining which modular components will be used in the variant design. Candidate designs are synthesized by pairing combinations of modular component alternatives with values of high level design variables. The design variables are passed as parameters to drive updates in the model skeleton. This flow of data is depicted in Figure 7.

For simple systems composed of only a few components, a full-factorial search of component combinations in the design space is possible. For more complex systems, appropriate discrete-variable optimization techniques can be used. It is important to note that the problem is likely to be multi-objective and thus will require a multi-objective optimization routine such as NSGA-II [21] coupled with a multi-attribute decision making technique. This is the approach taken in the developments in Ref. [9].

**FIGURE 7. DATA FLOW BETWEEN MODEL-BASED DESIGN TOOLS**

Detailed design modeling tools are responsible for translating design parameters and the chosen component alternatives into 3-D representations of parts with enough detail that they can be manufactured. Part geometry should be drawn as parametrically as possible, meaning that key dimensions are left as parameters which are referenced by lower level dimensions and geometric features. For example, the overall length of a multirotor arm is left as a parameter. This parameter is referenced by the arm tube and the motor mount pad features to size each and precisely locate their intersection.

The detailed design tools contain more than just 3-D models. They must be paired with logical rules which enforce design logic during model updates. These rules can be in the form of conditional statements, checks, or iterative loops. An example is a check on the wall thickness of a cantilever beam feature. The bending moment at the root of the beam will vary with beam length. After a change in beam length, a check rule implemented in code computes whether the thickness of the beam is enough to handle the new bending moment and increases the thickness if necessary. Logical rules are the mechanism by which impossible or invalid geometries are avoided. Several iterations of design and testing the rules will likely be necessary to achieve a working rule set. A range of 3-D modeling techniques can be used to avoid invalid geometries. A detailed exploration of this aspect of this work can be found in Ref. [22].

A concurrent design task is to identify the automated manufacturing techniques or manufacturing processes that will be used to manufacture each part. The manufacturing processes available may be subject to equipment or logistics constraints. Furthermore, each manufacturing process constrains the geometry of any part that it is used to produce. A relevant example is that 3-D printed parts with sharp corners have very high stress concentrations. Sharp edges should be filleted to prevent rapid failure of the part. On the other hand, laser cutters remove material during cutting operations, leading to a dimensional deviation from the CAD model. As such, each part must be designed with a manufacturing process in mind.

Throughout the modeling and design process, it is vital that the models for the vehicles and individual parts are validated for the range in which they will operate. For the purposes of ADAPt Design, the model must be validated across the whole expected range of the design.
CASE STUDIES

ADAPt Design was developed around a multirotor SUAS platform and applied to a fixed wing platform to assess its extensibility. This section illustrates ADAPt Design by following the development of these platforms starting from requirements analysis and ending in a linked set of model-based design tools.

Requirements Analysis

ADAPt Design is intended to equip Soldiers with one-off systems tailored to squad level mission needs. Stakeholders in this scenario are the Army personnel responsible for planning and executing ISR missions and the personnel manufacturing the SUAS. An assessment of the technical skill levels across these stakeholders drives a need for providing users with a small set of inputs that can fully capture the mission, without requiring detailed knowledge of design or aeronautics. Intuitive mission requirements such as payload type, range, endurance, speed, and size are chosen as inputs. To meet the on-demand vision and fill the capability gaps left by SUAS currently in the Army inventory, a need is established for converting inputs to a functional design in less than 48 hours.

Previous efforts have identified five representative mission profiles where SUAS could provide support to squad level operations: convoy surveillance and defense, perimeter surveillance and defense, building interior reconnaissance, cave interior reconnaissance, and jungle reconnaissance [8],[9]. The capabilities desired in these missions are addressed by a SUAS that carries one of four payloads: a video camera, communications equipment, LIDAR, or a target designator.

At the bottom of Figure 9, all of the components identified to fulfill the platform functions are classified as either modular or scalable. For the two platforms considered, the components common between the platforms happen to be more readily obtained off-the-shelf than fabricated. For this example, the shared components exactly coincide with the modular components and those unique to each platform coincide with the scalable components.

A component library is now built to document the components identified. The relative simplicity of the platforms and small number of components to track permits the use of the Excel® spreadsheet application for this task. Table 1 shows two sample component library entries in order to illustrate the type of information being tracked. As the product family becomes more developed, the library will be populated with alternatives of each component. Component alternatives are distinguished from one another by differences in their attributes. For example, two instances of propellers may be populated into the library: one with a 10 in. (254 mm) diameter and the other with a 12 in. (305 mm) diameter.

<table>
<thead>
<tr>
<th>TABLE 1. SAMPLE ENTRIES IN THE COMPONENT LIBRARY FOR A MODULAR AND A SCALABLE COMPONENT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Component:</strong> Motor</td>
</tr>
<tr>
<td><strong>Attributes</strong></td>
</tr>
<tr>
<td>1) Manufacturer</td>
</tr>
<tr>
<td>2) Model</td>
</tr>
<tr>
<td>3) Kv rating (RPM/Volt)</td>
</tr>
<tr>
<td>4) Weight (lbs.)</td>
</tr>
<tr>
<td>5) Body diameter (in.)</td>
</tr>
<tr>
<td>6) Body height (in.)</td>
</tr>
<tr>
<td>7) Shaft diameter (in.)</td>
</tr>
<tr>
<td>8) Shaft height (in.)</td>
</tr>
<tr>
<td>9) Bolt pattern small diameter (in.)</td>
</tr>
<tr>
<td>10) Bolt pattern large diameter (in.)</td>
</tr>
<tr>
<td>11) Base pad diameter (in.)</td>
</tr>
<tr>
<td>12) Bolt thread diameter (mm)</td>
</tr>
</tbody>
</table>

FIGURE 8. SUAS PERFORMANCE REQUIREMENT METRICS AND THEIR EXPECTED RANGE OF VALUES

Architecture Selection

A historical search approach yields two types of existing SUAS platforms that are simple, well understood, and cover all of the desired capabilities. These platforms are a multirotor for operations such as reconnaissance of building interiors or caves which require hovering and maneuvering, and a hand launched fixed wing SUAS to cover long endurance and convoy support type missions. A functional decomposition of each platform yields the functions in the left columns of each box in Figure 9.
The ranges of the mission requirements in Figure 8 permit a rough sizing exercise of both multirotor and fixed wing platforms. The result of this exercise is a first guess at the size of propellers, motors, and batteries the vehicles will have. This in turn gives an indication of the size of the speed controllers, central hub and arms for the multirotor, and wing, fuselage, and flight control surfaces for the fixed wing. Candidates for multirotor components identified by this analysis are organized into a matrix of alternatives shown in Figure 10. Speed controllers, GPS, and flight controllers have been omitted from Figure 10 because speed controllers match one-to-one to motors and the same GPS and flight controller are used across all variants.

The component information in Figure 10 allows the first configuration layout to be generated for each platform in the form of model skeletons. The model skeleton for the multirotor is implemented in CATIA® V6 and is shown in the left side of Figure 6. The multirotor skeleton is populated with locations of the interfaces between parts such as where the arm meets the hub, and also with environmental boundaries such as the plane where the ground exists when the vehicle is not airborne. Capturing as much information in the model skeleton as possible is beneficial as it will be leveraged by logical rules to enforce design logic. For example, the ground plane is later used to check whether the vehicle will be stable and sit level when it is on the ground.

### Modular Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Multirotor arm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attributes</strong></td>
<td><strong>Interfaces</strong></td>
</tr>
<tr>
<td>1) Length (in.)</td>
<td>a) Hub connection</td>
</tr>
<tr>
<td>2) Weight (lbs.)</td>
<td>b) Motor mount</td>
</tr>
<tr>
<td>3) Material volume (in.³)</td>
<td></td>
</tr>
<tr>
<td>4) Base height (in.)</td>
<td></td>
</tr>
<tr>
<td>5) Base width (in.)</td>
<td></td>
</tr>
<tr>
<td>6) Motor mount bolt hole diameter (mm)</td>
<td></td>
</tr>
</tbody>
</table>

### Scalable Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload</td>
<td>Video feed</td>
</tr>
<tr>
<td>Motor</td>
<td>Propulsion power supply</td>
</tr>
<tr>
<td>Arm</td>
<td>Throttle signal</td>
</tr>
<tr>
<td>Hub</td>
<td>Motor drive power</td>
</tr>
<tr>
<td>Propeller</td>
<td>Propeller mount on motor</td>
</tr>
<tr>
<td>Motor mount on arm</td>
<td></td>
</tr>
</tbody>
</table>

### Interface Design

The required components identified by the functional decomposition for the multirotor platform interface with one another in various ways. The network of interfaces between these components is depicted in Figure 11. Standards for each interface are developed as follows.

The modular components include the motor, propeller, battery, flight controller/GPS, servo motors, and speed controllers. By definition, these parts will not be modified during the design process. Parts interfacing with these components must conform to their pre-established interfaces. Several examples are the motor mount bolt geometry, the motor’s electrical connectors, the motor shaft-propeller connection, and the propeller swept disc area. The motor mount bolt geometry consists of four M3 screws positioned around a central shaft hole. All motors share this pattern but the spacing between the holes varies motor to motor. The standard for the motor-arm interface geometry (Figure 11, interface 10) is therefore defined as the pattern of four bolt holes visible in Figure 12. The spacing between the bolt holes is left as a parameter so that the design can be updated when a different motor is selected. The motor’s electrical connectors are of a specific type, size, and shape. This sets the standard for the motor-speed controller electrical interface (Figure 11, interface 8). The motor shaft is circular and has a specific diameter. Thus, the standard interface geometry between the propeller and motor (Figure 11, interface 9) is defined as a circle of that diameter. The propeller sweeps out a disc of diameter equal to the propeller’s diameter. This forms an interface with the hub (Figure 11, interface 3) in that the propeller must be far enough away from the hub, with some margin, to prevent interference between the parts. Additionally, parts mounted on the hub cannot overhang its edge into this swept disc.

![FIGURE 11. MULTIROTOR PLATFORM INTERFACES](image)

Other interfaces have degrees of freedom left to the designer. Design automation will be faster and encounter fewer errors if the number of degrees of freedom is reduced. This is accomplished through defining custom interface standards. An example is the arm-hub interface geometry (Figure 11, interface 4). This interface design is geometric in nature and is left totally to the designer. A custom standard of two circular aligning tabs and a single through-hole for a bolt is defined and is visible in Figure 12. All variant designs conform to this standard, but the spacing of the tabs is left as a degree of freedom. The spacing will be automatically scaled for each new design to match the width of the arm’s base.

![FIGURE 12. EXPLODED VIEW SHOWING GEOMETRIC INTERFACES ON A MULTIROTOR PLATFORM](image)
The fixed wing platform's fuselage, wing, and flight control subsystems are decomposed into several individual components. The configuration calls for parts assembled around a carbon fiber tube, which makes up the fuselage shaft. A battery cage, motor mount, component mounting plates, wing, and empennage mount slide onto the shaft and lock onto one another via alternating teeth. The geometric interfaces structure is shown in Figure 13. The electrical and logical interfaces are the same as the multirotor’s, but with the addition of servo motors to drive the flight control surfaces.

The fixed wing motor mount, shown in Figure 14, is an example of a modular component-derived interface standard. It consists of two parts. A motor-specific adapter plate mates to a mount attached to the fuselage shaft via four additional screws. This multi-part assembly converts interface geometry of any motor to a common geometry usable in all variant designs.

Concept Refinement and Design

As a stakeholder, the REF presents a manufacturing constraint which limits part sizes. The print bed tray size of the 3-D printer to be used is limited to 8 in. by 6 in. by 6 in. (203 mm by 152 mm by 152 mm) width x length x height. Another consideration associated with the 3-D printer is the print direction. Bending strength is degraded for bending displacements in and out of the width-length print plane. Thus, a part printed with a load bearing feature primarily in the width-length print plane.

The highest ranked combination becomes the design variant by default. However, the user is able to select a different combination if desired. The variant’s component combination and its design parameters are then passed to the detailed design tools. The tools are implemented in CATIA® for 3-D modeling and CATIA's KnowledgeWare® toolset to encode logical rules. The logical rules first search a repository of 3-D models to find and insert the selected instances of modular components into the main 3-D model. Then, driving design parameters such as arm length, hub width, and number of hub layers are pushed to the model skeleton, which is updated accordingly. At this point, logical rules parse the model, performing operations such as enforcing design logic that eliminates invalid geometry, scaling interface geometry to match the modular components, repositioning parts to fit on the hub, modifying structures to improve strength and save weight, and filleting 3-D printed parts as required for manufacturing. The multirotor arms and the fixed wing’s wing sections, control surfaces, and component mounts/adapter plates are designated as 3-D printed components.

Figure 16 illustrates the multirotor model before (top) and after (bottom) a design update. The top multirotor is the model in a default state, designed for generic requirements. Inputting new requirements into the design tools immediately triggers the tools to find different
modular components, resize the design parameters, and update the 3-D model resulting in the design shown on the bottom of Figure 16.

FIGURE 16. DEPICTION OF AN AUTOMATED MULTIROTOR MODEL UPDATE AS A RESULT OF A CHANGE IN REQUIREMENTS

Figure 17 shows a similar update for the fixed wing SUAS. Control surface span and chord, wing span, length, and airfoil, and battery cage dimensions are scalable in this platform.

FIGURE 17: DEPICTION OF A FIXED WING MODEL UPDATE AS A RESULT OF A CHANGE IN REQUIREMENTS

Table 2 compares the predicted values for each metric of the short range reconnaissance mission design (reproduced from Table 2) to the values obtained by building and testing the design. Figure 18 shows a flight test of this design with a still frame taken from the GoPro® camera feed. Figure 19 shows a fixed wing SUAS built for a similar reconnaissance mission. The results in Table 4 show that the ADAPt Design approach produced a design that met all the geometric and performance requirements. The performance of the as-built SUAS was conservative in that both weight and endurance exceed the requirements and the SUAS was able to perform aggressive flight maneuvers while carrying its camera payload. This is by design – models were built with conservative margins to avoid producing a SUAS that failed to meet mission requirements. However, the scale of the deviations between predicted and actual performance highlight a limitation of ADAPt Design, which is that it relies on very high modeling accuracy. The endurance model used is derived from a first-principles energy balance and a simple hover-only mission model. Build time is underestimated due to underestimating the time required to dissolve the specific type of 3-D printed support material used. In both cases, what may seem like insignificant assumptions or inaccuracies in modeling result in large discrepancies between predicted and actual performance of the as-built vehicle.

Method Validation

Validation of the method was achieved by inputting mission requirements into the design tools and manufacturing and testing the resulting design. The final products were inspected and flight tested so that their characteristics and performance could be compared to the input requirements. Two multirotor designs were produced: one for a short range reconnaissance mission and another for a payload ferry mission. The requirements for each mission are given in the “Target Value” columns of Table 2 and Table 3 respectively. The “Returned Design” columns are the values predicted by the design tools for the resulting design. Two fixed wing SUAS were also produced for similar missions using requirements specific to the fixed wing platform.

### TABLE 2. SHORT RANGE RECONNAISSANCE MISSION REQUIREMENTS AND RESULTING DESIGN VALUES

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Target Value</th>
<th>Returned Design</th>
<th>Returned 3-D Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. Outer Dimension (in.)</td>
<td>33.0</td>
<td>29.7</td>
<td></td>
</tr>
<tr>
<td>Max. Weight (lbf.)</td>
<td>5.0</td>
<td>3.08</td>
<td></td>
</tr>
<tr>
<td>Min. Endurance (minutes)</td>
<td>10.0</td>
<td>12.1</td>
<td></td>
</tr>
<tr>
<td>Max. Build Time (hrs.)</td>
<td>22.0</td>
<td>18.0</td>
<td></td>
</tr>
<tr>
<td>Payload Capacity (lbf.)</td>
<td>0.0</td>
<td>0.99</td>
<td></td>
</tr>
<tr>
<td>Sensor</td>
<td>GoPro®</td>
<td>GoPro®</td>
<td></td>
</tr>
</tbody>
</table>

### TABLE 3. PAYLOAD FERRY MISSION REQUIREMENTS AND RESULTING DESIGN VALUES

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Target Value</th>
<th>Returned Design</th>
<th>Returned 3-D Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. Outer Dimension (in.)</td>
<td>50.0</td>
<td>34.1</td>
<td></td>
</tr>
<tr>
<td>Max. Weight (lbf.)</td>
<td>20.0</td>
<td>5.16</td>
<td></td>
</tr>
<tr>
<td>Min. Endurance (minutes)</td>
<td>7.0</td>
<td>7.18</td>
<td></td>
</tr>
<tr>
<td>Max. Build Time (hrs.)</td>
<td>40.0</td>
<td>33.1</td>
<td></td>
</tr>
<tr>
<td>Payload Capacity (lbf.)</td>
<td>4.0</td>
<td>8.63</td>
<td></td>
</tr>
<tr>
<td>Sensor</td>
<td>none</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 4. COMPARISON BETWEEN REQUIRED, PREDICTED, AND MEASURED REQUIREMENTS FOR THE SHORT RANGE RECONNAISSANCE MULTIROTOR SUAS

<table>
<thead>
<tr>
<th>Requirement Metric</th>
<th>Required</th>
<th>Predicted</th>
<th>Actual</th>
<th>Error (predicted vs. actual)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outer Dimension (in)</td>
<td>≤ 33.0</td>
<td>29.7</td>
<td>29.7</td>
<td>0.0%</td>
</tr>
<tr>
<td>Weight (lbf.)</td>
<td>≤ 5.0</td>
<td>3.08</td>
<td>2.99</td>
<td>3.0%</td>
</tr>
<tr>
<td>Endurance (minutes)</td>
<td>≥ 10.0</td>
<td>12.1</td>
<td>15.1</td>
<td>19.9%</td>
</tr>
<tr>
<td>Build Time (hrs.)</td>
<td>≤ 22.0</td>
<td>18.0</td>
<td>20.5</td>
<td>12.2%</td>
</tr>
</tbody>
</table>

CONCLUSION

Modern military operations expose Soldiers to rapidly evolving and often unforeseen problems. The nature of these operations suggests that SUAS can provide crucial intelligence to Soldiers in a timely manner. However, equipping Soldiers with SUAS assets to meet unforeseen needs poses design and logistical challenges. A solution is to design and develop custom-tailored SUAS at the site of deployment. This on-demand approach is enabled by an automated SUAS design capability.

The focus of this work is to develop a framework to define and architect a set of product platforms that are highly compatible with design automation. The framework developed, called ADAPt Design, leverages concepts from product family design, reconfigurable system design, and systems engineering to enable on-demand design and production of SUAS. Applications to both multirotor and fixed wing SUAS prove the method’s ability to generate differing designs given contrasting requirements, and that designs meet their respective requirements. However, flight tests indicate that the design approach is in part limited by the accuracy of the underlying models. Future efforts will focus on improving the accuracy of the multirotor mission model to reduce the error margins observed in initial tests. Additionally, flight tests of the fixed wing SUAS designs will be used to validate the method’s ability to generate feasible designs of dissimilar platforms.

Even though ADAPt Design was developed around small systems, the method could be applied to architect adaptable subsystem designs within more complicated products. The authors believe that the method is relatively scalable, and that it could be modified to account for increased product complexity.

REFERENCES


