Improving Development Visibility and Flow in Large Operational Organizations

Jo Ann Lane[1] and Richard Turner[2]

[1] University of Southern California, Los Angeles, CA, USA
[2] Stevens Institute of Technology, Hoboken, NJ, USA
jolane@usc.edu, rturner@stevens.edu

Abstract. In large operational systems, understanding the status of evolutionary capability development is often difficult. This is particularly true where capabilities depend on significant software components that are managed and operated as interacting subsystems. Schedules are rarely stable due to significant external drivers, thus integrated master schedules are hard to maintain and update. On-demand (pull) scheduling methods have been shown to smooth flow and maximize value across a process. The mechanics of these methods enhance visibility by forcing informed discussions on value, capability, and priority and by providing timely, relevant information to higher-level engineering organizations. This paper uses a notional information management system supporting a large health care system as an illustration of a management architecture that supports such an approach. The architecture includes a network of kanban-based scheduling systems, enhanced visualization, and employs a services approach to systems engineering that allows its work to be quantized as part of the overall development flow.

Keywords: systems engineering, software engineering, kanban, pull scheduling, systems of systems, management visibility.

1 The Story So Far…

In the summer of 2011, the US Department of Defense presented the Systems Engineering Research Center with a critical problem: how to replace complex integrated master schedules and plans and provide more effective management within a large, evolving operational system of systems. Additional concerns included the inability of scarce systems engineering (SE) resources to support schedule-driven projects, decisions made late or at a level removed from the context, and a lack of visibility into the status of system-wide capability developments—similar to issues already documented by a defense industry organizations [1]. Lean approaches based on Deming and The Toyota Production System have been deployed in manufacturing for decades and are well documented in the business and academic literature. In the last few years, however, on-demand scheduling techniques, such as the lean practice of *kanban*, have been successfully modified and adapted to support more agile, value-based scheduling in managing software projects [2, 3, 4, 5]. After considering the

B. Fitzgerald et al. (Eds.): LESS 2013, LNBIP 167, pp. 65–80, 2013.

issue, we decided to investigate the compatibility of the lean concepts with the sponsor's needs. Could they be applied to managing systems at a larger scale such as complicated systems of systems, and could they be used for systems engineering activities both at the portfolio and project levels of abstraction? Could they also be used across contracts, where non-governmental organizations were developing individual projects without close inter-project communication?

The result was an investigation into lean management, and a series of thought and simulation experiments on how to scale the kanban concept [6]. These in turn led to the need to consider systems engineering as a service, understand how communication between kanban systems could be accomplished, how to establish the comprehensive and current values for work items, how resources could be shared across kanban systems, and what types of controls were best to manage flow and protect resources [7].

It became apparent that discussions, arguments and whiteboard work were not converging, and it was necessary to apply the ideas to a specific, real environment. This paper describes an architecture for managing flow through a highly specialized information system of systems (SoS) as represented by a large health care system.

2 Concerning Kanban-Based Scheduling System Networks

In [6], we created the fundamental building block of the architecture – The Kanban-based Scheduling System (KSS). Illustrated in Figure 1, the KSS is designed to be replicable as a single system (made up of multiple KSS building blocks) or a network of such systems. Its functional requirements were to:

- Coordinate multiple levels of development activity
- Support analysis and decision making at every level
- Flexibly schedule work considering value across the system of systems
- Balance work in progress (WIP) across resources with organizational capacity to improve flow
- Make visible to all levels progress of capability development and deployment
- Establish a basis for continuous improvement in a rapidly changing environment

The fundamental concept is that each organizational entity adopts a hybrid system and organizational value-driven KSS designed by the organization to meet their specific needs. Each KSS includes a kanban board with specific and public management controls designed to integrate the KSS with others. The kanban board is a working tool used to track work and collect typical statistics such as cumulative flow diagram information. An organization with many sub-organizations may choose to have a dashboard that rolls up the information from the LSSs within it (or tasked by it) into a more informative visual tool. Such dashboards act as information radiators at all levels of development activity. The result of the system is that each KSS provides current, consistent information that flows up, down and across the organizations as needed.

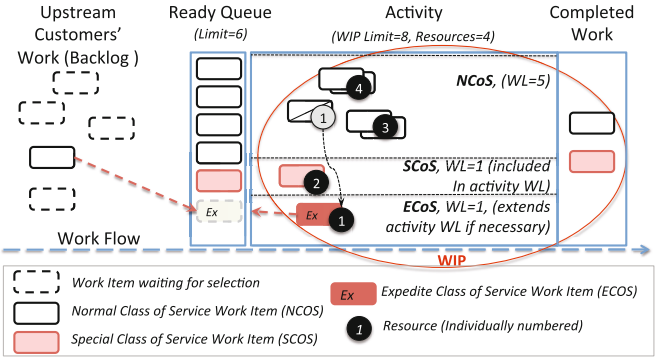


Fig. 1. The KSS Building Block [6]

Each KSS has a single backlog of unlimited length and a single acceptance queue with a capacity-informed limit. Each also has a queue where completed items are reacquired by the source KSS or forwarded to a downstream KSS. The organizational KSS may include lower-level KSSs if appropriate. Table 1 shows the template designed to characterize a KSS within the network.

Table 1. KSS Summary Template

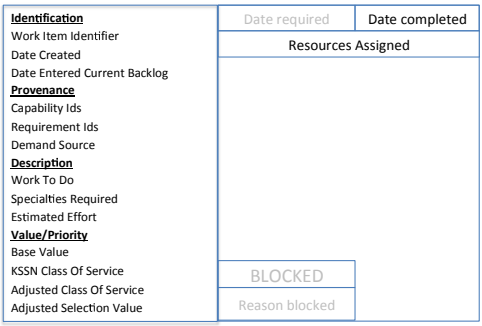
KSS Name			
Demand:			
Work sources	Organizations that can assign work items to the KSS		
Resources:			
Dedicated	Resources under control of this KSS		
Shareable	Resources available to share on teams with other orgs		
Sourced	Organizations (KSSs) to which work items can be assigned		
Managed resources	Any specialists that are managed individually		
Activities:			
Description	WIP Limit	Resource Type	Cohesion
		Internal, Sourced, or X-discipline team	Interruptible or Must complete
Flow and Visibility:			
Additional CoS handled	CoS beyond the system-wide that are recognized by this KSS		
Additional CoS introduced	CoS defined for work this org assigns to other KSSs		
Work Selection Value Adjustments:			
Source-based	CoS-based	Resource-based	Completion-based
Goals	From GQK analysis		
Questions answered	From GQK analysis		
Data maintained/used	From GQKanalysis		
Information shared	e.g. Avg. Lead time, Avg. blocked tasks. Avg. time blocked		

2.1 Work Items

In most systems of systems, capabilities are identified at a high level. Systems engineering decomposes these into requirements, which are further decomposed into

software features or hardware components. Any of these can be considered a work item. Work items flow through the system, and are characterized as shown in Table 2. A work item is created within a KSS, assigned certain attributes – class of service, value, associations – and may flow from KSS to KSS until it is complete and then either returns to the source KSS or notifies the source KSS and passes downstream. Work items may be decomposed, assigned to (or selected by) individual or multiple resources, and are pulled through the individual KSS in a normal kanban board fashion. While each work item has information that is carried along with it, its value is dependent on many factors – local and systemic. It may be associated with other work items in such a way as to comprise a higher level accomplishment – for example several work items assigned to different KSSs may actually make up the necessary features to meet a particular requirement.

Table 2. Generic Work Item



2.2 Network Flow

Work items flow by negotiation between the KSSs, with each organization agreeing to what is essentially a service contract for the work item. Backlog mechanics operate with mutually agreed to rules as to when and how prioritization and selection take place. Value parameters and functions, Classes of Service, and service level agreements may be established as network wide or KSS controls. Network-wide controls generally take precedence over local controls. The network also communicates the status of each work item as it flows through the system.

It should be noted that this is not a value stream analysis nor model. It is, however, a scheduling system that if used correctly, may identify the type of information often sought in such analyses from the current data on the kanban boards and dashboards.

Because kanban concepts have been primarily used with single level value streams, we wanted to understand the information needed for decision making, including scheduling and flow monitoring/control, at each level of SE activity or utilization. This would allow us to construct a KSS that would support visualization of WIP and status for each specific level. It would also provide insight into the information flow required. To accomplish this, we turned to the Basili, Caldieri, and Rombach Goal-Question-Metric approach [8, 9]. For each level we defined the goals and the

questions that made sense to ask in order to determine if the goals were being met. Given our research is to investigate KSSs, we decided to fully utilize the metrics available from flow and pull concepts. To acknowledge this, we affectionately referred to the results as goal-question-kanban (G-Q-K) information.

2.3 Systems Engineering as a Service?

The idea of applying SE as a service within an on-demand scheduling system is not as farfetched as it may seem to some systems engineers. It effectively merges the SE flow and the software development project flow. These services act as any other work item in the KSS. SE performs early definitional activities, like operational concepts, architecture, and functional allocation. Other activities are ongoing like incremental verification and validation. Still others are performed at the request of a project and include trade studies, specialty engineering (like safety or security), and impact assessments. All involve maintenance and evolution of long-term, persistent artifacts that support development across multiple projects.

With the knowledge in these persistent artifacts, SE can be opportunistic in applying its cross-project view and understanding of the larger environment to specific projects individually or in groups. It can also broker information between individual projects where there may be contractual or access barriers. When a system-wide issue or external change occurs, SE can ensure that the broader issue is handled in an effective and compatible way [7].

3 The Health Care System Environment

We decided to use a health care system as the target for our prototypical architecture because of its similarity to many operational defense and intelligence community systems. The target system is a set of integrated medical information management systems. It consists of hardware, over two million lines of source code, numerous commercial-off-the shelf (COTS) software products and communications networks. Its primary goal is to support the administration and delivery of health care in networked set of hospitals and clinics in a timely and safe manner, coordinating a variety of providers and specialists. Key overarching requirements are to ensure patient safety and protect patient information according to regulations.

The Health Care Development Organization is around 1000 engineering professionals, some of whom are out-sourced, consists of three groups. The *systems engineering group* performs analyses related to new or enhanced capabilities, requirements development and allocation, evaluations of medical devices for integration, system performance assessments and upgrade recommendations, deployed and development networks, specialty engineering, SoS-level integration and acceptance testing. *Product teams* are responsible for software maintenance and enhancement for the custom constituent systems or products; database structures and embedded procedures, COTS product tailoring, integration, and upgrades; licensed data upgrades such as pharmacy approved formularies; and, responding to issues

beyond the scope of the user help desk. The *user support group* runs the help desk, site configuration management, and site installations and upgrades.

Key custom software includes user access management, patient management, pharmacy, laboratory, radiology, and patient telemetry. The constituent systems share a single database that maintains the information for all of the patients and personnel related to a given health care site. There are also interfaces to other health care systems, including custom legacy systems, COTS products, and electronic medical devices such as heart rate monitors and infusion pumps.

The current systems engineering and software engineering organizations are fully staffed with respect to development budget. When new needs or capabilities are identified, systems engineering analyzes the new needs/capabilities in terms of the given systems and decides how address them. Often multiple new needs/capabilities are analyzed together to facilitate the identification of common solutions that can support more than one need/capability as well as support performance upgrades and technology refresh. The results of the analysis activities are a set of requirements. The next step in the process is to allocate those requirements to one or more products for implementation. Figure 2 provides an example that illustrates how multiple requirements are derived from one or more needs and then mapped to the enterprise products for implementation.

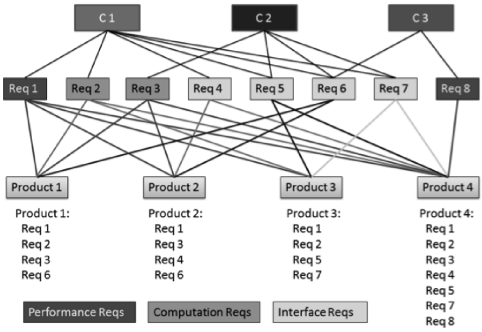


Fig. 2. Capabilities to requirements to products

Once the requirements are allocated to the products, the product teams analyze them and convert them into features and stories for implementation. Systems engineering monitors the capability "pieces" to guide their system integration and testing activities. When all of the capability requirements are implemented in the affected products and deployed, the mission capability is considered "completed."

Several issues exist. There is no visibility at the capability level showing which user stories are related to which capabilities and which products are implementing pieces of the capability. The systems engineering resources are hampered by variable, multiple tasks, and rapidly changing priorities. Software tasks become blocked waiting for systems engineering tasks to complete. As a result, started tasks are difficult to complete in a timely manner.

4 The Health Care KSS Network

The KSS Network prototype defines a 3-tiered management architecture:

1. Executive/Stakeholder Management (ESM)
2. Capability Engineering (CE)
3. Product/Domain Engineering (PDE)

Figure 3 provides an overview of the Health Care System KSS-network showing function levels, KSS elements for monitoring and control, and Dashboards for providing information from multiple KSSs.

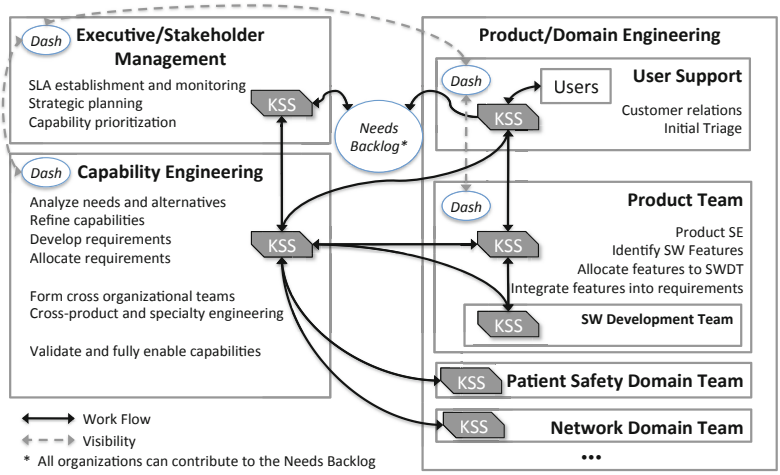


Fig. 3. Overview of KSS Network

Classes of service (CoSs) provide a variety of handling options for different types of work and affect the next work item selection value for KSSs. They may be aligned with Service Level Agreement priorities. Most CoSs are intended to ensure priority rather than force immediate execution. There are CoSs that are disruptive–that is, they can suspend current work in progress. These are associated with critical or expedited work to allow swarming of all appropriate resources to ensure completion as soon as possible. However, disruptive CoSs are minimized because they counter the normal kanban philosophy of completing work rather than interrupting it. While most CoSs are shared across the entire KSS network, individual KSSs may define additional *KSS-Specific* CoSs to handle flow specific to their types of work. Table 3 shows the COSs that apply to all the work in the KSS Network.

The calculation of the selection value of each work item depends on its inherent static value as assigned at it's creation, it's inherited value by being associated with the value the requirements and capabilities it supports, and the state of the development process (e.g. the status of capabilities, requirements and other work items). One simple calculation of this could be:

Vt = *Vinh* + *Vstat* +*Vbase*; where *Vinh* = Sum of all the current values of the requirements the work item supports (with each requirement also includes additional value for multiple capabilities that it supports); *Vstat* = Adjustments due to the status of work, such as additional value added for near completion of requirements or capabilities, or a negotiated value between the work item owner and the work provider (an example might be the reduction of value to group certain work items that require special handling like certification); and, *Vinit* is the initial value defined. Regardless of the value, the item's current class of service controls the rules by which the work item may be selected.

Table 3. General Classes of Service

CoS	Description
Critical Expedite	Safety, security, or other emergency work items. Disruptive: requires necessary resources to stop current work and complete it.
Important	Very high priority work items such that this work takes priority over other work in the ready queue. Not Disruptive.
Date Certain	Work items that must be completed by a specific date or there will be significant consequences.
Standard	The normal CoS for the development organizations work.
Background	Work that must go on but is usually not time critical. It includes things like architectural enhancements, low-level technical debt, or research and environmental scanning

5 KSS Descriptions

Each KSS is based on the workflow, the G-Q-K information, and the special circumstances and needs of each organization of resources represented by the KSS. There are nearly as many ways to define a KSS as there are to define a system. We simply recommend processes and visualizations appropriate to our target organization. Each description includes a summary, process flow descriptions, and visualization tools.

5.1 Executive/Stakeholder Management (ESM)

The ESM level determines which proposed capabilities (or capability enhancements) are going to be approved for development. As part of this process, ESM assesses the value of the capability against its expected cost and schedule to develop. This highest-level in the KSS network is concerned primarily with the current status of identified capabilities (or needs) as represented by the development state of each "not fully deployed" but "approved for development" capability – essentially WIP. At this level, the KSS is tracking capabilities and their priority. The insight it provides should inform decisions about overall organizational strategy, resource staffing, and development funding priorities. Table 4 provides the ESM KSS Summary.

Accepting/Selecting Next Work Item. Requests for system capabilities come from the users, systems engineering groups, and strategic initiatives. There is always a

backlog of ideas needs, and wants. ESM must identify the highest priority capabilities. They must balance adding new capabilities with improving existing system capabilities and maintaining the infrastructure. They must also act on critical issues regarding patient safety, infrastructure failure, and regulatory changes. The outcome of this process is sending only the highest value and most critical work to the systems engineering group to analyze and develop.

Table 4. ESM KSS Template

Executive/Stakeholder Management KSS				
Demand:				
Work sources	Needs backlog, Stakeholders, Critical Events, Strategic Plans			
Resources:				
Dedicated	IT Managers, CTO, …			
Shareable	None			
Sourced	CE			
Managed resource specialties	None			
Activities:				
Description		*WIP Limit*	*Resource Type*	*Cohesion*
Capability Analysis			Sourced (CE)	Interruptible
Capability Prioritization-CoS Assignment			Internal	Must complete
Capability Development Project			Sourced (CE)	Interruptible
Flow and Visibility:				
Additional CoS handled	None			
Additional CoS introduced	None			
Work Selection Value Adjustments				
Source-based	*CoS-based*		*Resource-based*	*Completion-based*
None	None		None	None
Goals	G1. Deploy capabilities according to value-based priorities and CoS. G2. Understand source/cause of blocked work flows G3. Strategic IT decisions based on current and projected WIPs and backlogs (examples might include investments in additional resources (hardware, tools, people) or decisions to drop lower priority capabilities). G4. Changing needs and priorities are integrated with existing strategy			
Questions answered	Q1. What capabilities are currently in progress? Q2: What capabilities are currently blocked? Q3: What capabilities are pending acceptance? Q4. Are the planned and actual values of each deployed capability tracking? Q5: Are the current WIP level for ESM activities correct? Q6. What is the average time to completion for "accepted" capabilities by CoS? Q7. What is the requirements volatility by capability? Q8. What KSSs show capacity not meeting demand? Q9: What KSSs indicate excess capacity?			
Data maintained/used	KSS1: Flow data on CE and Product Teams* KSS2: Average time to deploy capabilities for each CoS priority level KSS3: Relationships between capabilities and requirements KSS4: Status of requirement completion/deployment KSS5: % of requirements completed/deployed for each in-process capability KSS6: Status of SE tasks supporting capability acceptance decisions *Includes CFD (throughput, WIP, Lead time), backlog level, resource utilization, blocked tasks, and similar data.			
Information shared	Capabilities under development, CFDs for each Capability, Network Value Tracking,			

Some work items initiated within the ESM level are special studies related to the prioritization of capabilities and the possible combination of multiple needs into a more effective capability need. This work includes cost and schedule estimations, Ops Concept development, COTS evaluations, and other traditional early SE activities.

Allocating Resources and Team Development. ESM must understand the overall capacity, work in progress, and resource distribution across CE and PDE teams in order to determine the highest priority capabilities and decide how to meet strategic needs and balance ongoing tasks. Starting too many capability developments can lead to less effective execution, while starting too few may jeopardize stakeholder satisfaction. This organization must work closely with the CE organization and User Support to map the landscape reflected in the needs backlog.

Completion and Disbursement. While the decision to deploy is a systems engineering or PDE decision, the declaration of a capability being "finished" (i.e. fully implemented and deployed) is usually reserved for the ESM.

KSS review at this level examines the work in progress, demand, capacity, and performance to ensure it is focused on achieving capabilities and handling critical events. Resource management, including budgeting, requires an understanding of how development resources are being utilized throughout the system, what is in the backlog of desired capabilities, and areas where there is excess capacity or capacity is insufficient for the projected demand. Budgeting is a factor in determining how much demand is realistic regardless of capacity. Strategic changes to resource mix across the SoS may be needed through hiring, contracting, or moving resources.

5.2 Capability Engineering (CE)

CE represents all capability-related SE activities, specialty SE support for product teams, including software system engineering tasks, where software is a key component in the requirements allocation. CE is responsible for creating capability descriptions that incorporate the needs identified and prioritized by the ESM level. CE must balance the various SE resources as they work with both internal activities and lead cross-organizational teams in CE-related activities. Decisions and scheduling of the SE resources must include front-end and ongoing architectural work as well as supporting development, integration, verification and validation with product teams.

The CE KSS represents multiple levels of activity and may choose to break into multiple KSSs as the complexity grows. However, the initial concept is a single KSS that handles a variety of different activities. First, the CE must respond to the ESM requests for analysis and SE support to ESM decision activities and for the development of capabilities that are the highest priority to the SoS. The CE also provides SoS analysis support to the various PDE Teams and manages the limited number of SoS specialty engineering resources. Given the goals associated with this level, both the kanban board and the dashboard will be somewhat "busy" in terms of information. Table 5 presents the CE KSS template.

Accepting/Selecting Next Work Item. As requests come in for systems engineering services, they are accepted, roughly estimated, possibly broken into smaller tasks, and valued. An additional CoS is assigned as necessary and then the work items are added

to the backlogs for the appropriate resource. Queue length limits are usually maintained for backlogs, and the level of the queue in terms of a percentage is a reasonable measure of demand.

Allocating Resources and Team Development. Many CE tasks will require a team with expertise in one or more specialty engineering areas or may require collaborative support from one or more PDE Team SEs. The CE negotiates with the appropriate teams for the specific resources they need. CoS, nearness to completion of the requirement, and other factors are considered. For requests from software teams, the special software CoS is applied as described in the summary. Capability Requirements Development work items are created, sourced to the various PDE Teams, and tracked to completion. Any negotiation required is accomplished before CE or the PDE Team accepts the work.

Completion and Disbursement. As CE completes ESM analysis work items, they are delivered directly to the ESM and identified as "done" on both the ESM and CE boards. Analysis tasks from PDTs are handled the same way. Work sourced to the PDE Teams may be completed and deployed by the PDE Team. The PDE Team will share data to update the CE KSS and Dashboard. There could be an activity to provide requirement completion verification and validation within the CE KSS, but in this initial concept, it is handled within PDE. Data is passed to the ESM dashboard.

KSS Review. Walking the CE KSS involves tracking the work in progress, identifying flow problems and blockages, resolving resource issues and blockages, and monitoring the demand queue so that when resources are available the next most valuable piece of work is accepted. The review tracks the WIP-level and demand for specialty resources to avoid blockage, overwork, or underutilization. Work items should be scanned for adjustment to work value or priority on completion-based criteria. Technical or PDE Team issues should be reviewed, and often it is good to include members of critical PDE Teams in the review.

5.3 Product/Domain Engineering (PDE)

At the PDE level, there are separate KSSs for each product or domain team in the enterprise. The PDE KSSs are similar to those used in many software development organizations today, with the added requirement for systems engineering within the product or domain scope. Constituent systems/products need to provide information to higher level KSSs and dashboards all the way to the ESM level.

The User Support (US) Team operates at the PDE level because it interfaces with the product and domain teams. There are occasions, however, when it influences the needs backlogs, or when it uncovers an issue (e.g. patient safety or privacy) that requires engagement with ESM and CE to handle the solution. Each product or domain team is responsible for responding to problems the US team can't handle.

Each product team creates its own organization. If outsourced, contractual requirements and its corporate governance influence the KSS implementation. For example, if the company operating the product team uses a matrix organization for SE, they may create a separate KSS for the SE resources that might cross product team boundaries. If the SE resources are each dedicated to a specific product, then their tasks can be included in the product or the software development KSS.

Table 5. CE KSS Template

Capability Engineering KSS				
Demand:				
Work sources	ESM, PDT, Internal			
KSS Resources:				
Dedicated	SoS SEs, Specialist SoS SEs (performance, algorithms, interface, security…)			
Shareable	Most			
Sourced	PDE Teams			
Managed resources	Specialty SoS SEs (performance, algorithms, interface, security…)			
Activities:				
Description		*WIP Limit*	*Resource Type*	*Cohesion*
Capability Analysis			X-discipline team	Interruptible
Operational Concept Development			Internal, X-discipline team	Interruptible
Capability Requirements Creation			Internal, X-discipline team	Interruptible
Capability Requirement Development			Sourced	Interruptible
Special Engineering Services			Internal (managed)	Interruptible
Flow and Visibility:				
Additional CoS handled	Software Service CoS: One of the issues identified was the amount of time product tasks were blocked waiting for SoSE (CE) support. This CoS is applied to all Specialty Engineering Services work items from PTs with significant software components. The CoS is not interruptible and provides a guaranteed WIP capacity. Resource reallocation is allowed to meet this CoS.			
Additional CoS introduced	None			
Work Selection Value Adjustments				
Source-based	*CoS-based*	*Resource-based*	*Completion-based*	
None	None	None	Value of work items associated with requirements or capabilities within 15% of completion are raised by 10% at selection cadence points	
Goals	G1. Cost-effective, timely alternatives identified for new capabilities/enhancements G2. Adaptable, flexible, multi-purpose solutions for new capabilities/enhancements G3. Specialty engineering responses to software teams' SE requests do not create excessive delays in capability development G4. Provide quick response to changing needs and priorities			
Questions answered	Q1. What work is currently blocked? Q2. What is the % of capabilities that are deployed within the desired timeframe? Q3. What is the predicted time to completion for "accepted" CE tasks (by class of service)? Q4. Where is capacity not meeting demand (by capability specialty engineering discipline)? Q5: Where is there excess capacity (by capability specialty engineering discipline)? Q6: What is the age of items in the CE backlog queues? Q7. What are the current CE WIP levels? Q8. What are the current CE backlog levels? Q9. What is the balance between CE WIP and CE backlog?			
Data maintained/used	KSS1: Number/status of tasks in product-level queues (analysis, backlog, WIP, blocked) KSS2: Number of tasks in product-level queues blocking other tasks (e.g., dependent tasks) KSS3: Relationships between capabilities, requirements, and features at product level KSS4: Percentage of each in-process requirement already completed/deployed KSS5: Average User Support request task completion time			
Information shared	Requirements allocation, status and deployment data; CE and PDE flow information			

User and site support personnel interact directly with the users and other operational stakeholders for the system of systems. They provide insight and triage for user requests; they aggregate and categorize desired capabilities or required maintenance actions, and forward them for resolution to the CE or PDE Teams as appropriate.

The *US KSS* is set up to manage the resources of the personnel handling the triage function and to identify critical issues rapidly. They track issues to completion and support information requests on the status of specific issues. This KSS is modeled on the system developed by Joshua Bloom at The Library Corporation, and the authors appreciate his support in this research. Table 6 provides the US KSS Template.

Accepting/Selecting Next Work Item. US is the connection between the development system and the user population. Many user calls do not require development and are managed through the US KSS alone. Tickets for problems that require technical development work are written up and entered into the KSS demand queue. Initial estimations are of the "t-shirt size" variety and tickets are classified according to product, domain or other attribute. Any tickets critical to patient safety or require expedited activity are immediately handed off to the ESM, CE, and PDE teams to swarm and resolve quickly. Otherwise, initial classes of service are assigned.

Allocating Resources and Team Development. Once a ticket is entered into the demand queue, it is determined to be product specific and sent to a PDE team, it is determined to involve multiple products/domains and is entered into the ESM needs backlog as a systems of systems capability issue, or, it is not immediately understood and so sent to the SoS team to analyze and recommend action. All such tickets are maintained in the KSS as in-process work and tracked through the system to completion so US can provide feedback on its status to users.

Table 6. User Support KSS Template

User Support KSS			
Demand:			
Work sources	User requests		
Resources:			
Dedicated	Help Desk Personnel, SW/System Engineers		
Shareable	None		
Sourced	PDE Teams, CE		
Managed resource specialties	SW/System Engineers may be handled as managed resource specialists		
Activities:			
Description	*WIP Limit*	*Resource Type*	*Cohesion*
Call Reception and triage		Internal	Must complete
Secondary ticket review		Internal	Interruptible
Ticket assignment		Internal	Interruptible
Flow and Visibility:			
Additional CoS handled	None		
Additional CoS introduced	None		
Work Selection Value Adjustments			
Source-based	*CoS-based*	*Resource-based*	*Completion-based*
None	None	None	None
Goals	Not yet addressed		
Questions answered	Not yet addressed		
Data maintained	Not yet addressed		
Information shared	Not yet addressed		

Completion and Disbursement. When PDE Team or CE development work is done, the US advises the ticket requestor(s) the ticket has been resolved and provides a resolution to the user: a software patch, workaround, or fix deployment date.

KSS Review is focused on the ability to effectively triage and assign tickets. Surveillance of the status of the technical work that entered through the US KSS provides a measure of response time to user requests and may be accompanied by user satisfaction information. Because of the rapidity with which most help desk activities occur, the dashboard provides the information of a kanban board.

The *PDE Product Teams* are responsible for one or more of the Health Care System products. The teams include systems engineers, specialty engineers, software engineers, hardware engineers, and often subject matter experts that support feature determination and development. System of system capabilities may require multiple product teams to create or enhance features, implement similar features in different ways, or collaborate to develop a common solution for the specific systems. If CE is the heart of the system of systems, the product team is the arms and legs.

A PT KSS is focused on maintaining the product at a high level of effectiveness and evolving it to support system capabilities as well as product capabilities. There is always some tension among the new feature development, older feature enhancement, and typical maintenance that is required in a technology and safety critical environment. The KSS uses the various CoS defined for the system to manage flow so that major capability developments proceed at a reasonable pace without significant impact on ongoing project level work. Table 7 provides the PT KSS Template.

Accepting/Selecting Next Work Item. Selection at this level is all about balancing: the capacity with the demand, new work with ongoing activity, and SoS value with product value. While selection decisions are supported by the inherited value determination and CoSs, the product teams still negotiate the flow. The sourcing customers and PT members look at the mix of tasks in the demand queue, evaluating each according to the system values, product values and resources available, as well as considering what items represent the final parts of a requirement or capability.

Allocating Resources and Team Development. Most of the PT work is performed by groups of resources, often in a multi-discipline project team. Individual SE resources must also be available to participate in the cross-discipline/cross-system teams used in the CE in capability analysis, so there may be a reason to apply some sort of Project-level CoS that reserves some capacity for supporting those activities.

Completion and Disbursement. Since PTs are responsible for integration, V&V and deployment, their kanban board addresses these activities. Data on status, acceptance and availability for inclusion of the various work items in completing capability implementation is always provided upstream to the sourcing KSS.

KSS Review. Walking the kanban board and reviewing the dashboard at the product level consists of looking for blocked work—resource conflict issues, sourcing delays, and rework are the main sources here. If the PT cannot complete work items within the established statistical bounds, changes must be made quickly to balance demand.

Table 7. Product Team KSS Template

Product Team				
Demand:				
Work sources	US, CE, Internal, other PDE Teams			
Resources:				
Dedicated	SEs, HW and SW developers			
Shareable	SEs			
Sourced	SW Developers (SDPT), Specialty Engineers (CE), Domain Specialists			
Managed resource specialties	Varies by team			
Activities:				
Description		*WIP Limit*	*Resource Type*	*Cohesion*
Requirements analysis & feature definition			Internal, X-discipline team	Interruptible
Feature development and integration			Internal, Sourced	Interruptible
Requirements V&V			Internal, Sourced	Interruptible
Deployment			Internal, Sourced	Must complete
Flow and Visibility:				
Additional CoS handled	Software Service CoS: One of the issues identified was the amount of time product tasks were blocked waiting for SoSE (CE) support. This CoS is applied to all Specialty Engineering Services work items from software PTs. The CoS is not interruptible and provides a guaranteed WIP capacity. Resource reallocation is allowed to meet this CoS.			
Additional CoS introduced	Certification required – Applies where work is bundled to prevent costly recertification.			
Work Selection Value Adjustments				
Source-based	*CoS-based*	*Resource-based*	*Completion-based*	
Varies by team	Varies by team	Varies by team	Support to work associated with requirements or capabilities within 15% of completion are raised by 10% at selection cadence points	
Goals	G1. Capability-allocated requirements are developed and deployed according to value G2: Product requirements/features allocated to increments and spins based on value G3. Product team responds quickly to changing product needs and priorities G4. Minimize workflow disruptions in product increments and spins G5. Minimize rework due to poorly understood capability requirements G6. Product team provides timely responses to user support issues/problems			
Questions answered	Q1. Value of product-level work currently blocked? Q2. What is the % of requirements completed within the desired timeframe? Q3. Where is PT capacity not meeting demand? Q4: Where is there excess PT capacity? Q5: How often is the average item age in product backlogs outside expected levels? Q6. What are the current product-level WIP levels? Q7. What are the current product-level backlog levels? Q8. What is the product-level response time to SW requests?			
Data maintained	KSS1: Flow data on Product Team* KSS2: Number/status of tasks in demand queues KSS3: Number of tasks in product-level activities that are blocking other tasks KSS4: Relationships between capabilities, requirements, and features at product level KSS5: Percentage of each in-process requirement already completed/deployed KSS6: Average User Support request task completion time **Includes CFD (throughput, WIP, Lead time), backlog level, resource utilization, blocked tasks, and similar data.*			
Information shared	Flow data on Product Team*			

6 Conclusions and Further Research

Much of this work has been engaged in thinking through the various scenarios that exist in highly complex system development, sustainment and evolution. The team is convinced of the validity, and is moving forward with the research, pending further funding. We have begun to develop simulations of this KSS instantiation, and have initiated conversations with major aerospace companies to support a multi-stage analysis, investigation and piloting of the concept.

Acknowledgements. This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. Addressing the problems also required support from practitioners and experts. A volunteer industrial working group provided their experience to the research. Of particular value were David Anderson, Curt Hibbs, Suzette Johnson, Don Reinertsen, and Jim Sutton.

References

1. NDIA-National Defense Industrial Association. Top Systems Engineering Issues In US Defense Industry. Systems Engineering Division Task Group Report (September 2010), http://www.ndia.org/Divisions/Divisions/SystemsEngineering/Documents/Studies/Top%20SE%20Issues%202010%20Report%20v11%20FINAL.pdf
2. Poppendiek, M.: Implementing Lean Software Development. Addison Wesley, Boston (2007)
3. Larman, C., Vodde, B.: Scaling Lean & Agile Development. Addison Wesley, Boston (2009)
4. Anderson, D.: Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press, Sequim (2010)
5. Reinertsen, D.G.: The Principles of Product Development Flow. Celeritas Publishing, Redondo Beach (2010)
6. Turner, R., Madachy, R., Ingold, D., Lane, J.: Improving Systems Engineering Effectiveness in Rapid Response Development Environments. In: Proceedings of the International Conference on Software and System Process 2012 (2012)
7. Turner, R., Madachy, R., Ingold, D., Lane, J., Anderson, D.: Effectiveness of kanban approaches in systems engineering within rapid response environments. In: Proceedings of the Conference on Systems Engineering Research 2012, Procedia Computer Science, vol. 8. Elsevier (March 2012)
8. Basili, V., Weiss, D.: A Methodology for Collecting Valid Software Engineering Data. IEEE Transactions on Software Engineering 10(3), 728–738 (1984)
9. Basili, V.R., Seaman, C.: Metric-Based Quality Management. Software Engineering for Embedded Systems Series, Fraunhofer IESE, Kaiserslautern, Germany (2010)