25
Application of Systems Engineering to Safety and Risk Management: a Human–Systems Integration Perspective

Tareq Ahram and Waldemar Karwowski

25.1
Introduction

This chapter starts by discussing concepts and tools from systems engineering (SE), namely human–systems integration (HSI), focusing on the direct benefits that SE and HSI offer to managing risk and subsequently loss.

Most systems have an inherent risk associated with them that can be calculated using reliability engineering methods such as failure modes and effects analysis (FMEA) or data-driven reliability calculations, based on probability and statistics. Employing these data, system variables such as mean time between failure (MTBF) and system availability can be calculated to a reasonable tolerance.

The most variable component in these systems, however, is the human element. Humans exhibit behavioral characteristics driven by emotions, mood, activity level, fatigue, and any number of biological factors. That said, humans are also highly adaptable and flexible, sometimes serving as saviors of a system and not just a loose, uncontrolled element.

HSI investigates human strengths and weaknesses from a body of growing knowledge called human factors (HFs) and pair these dynamics with system features that are complementary. For example, humans with their large associative memories and seemingly limitless data storage capacities are adept at pattern recognition, be it speech, faces, or distorted images. Physical computer systems currently are not as capable in this regard; however, they outperform humans in complex computational capacities.

Joining these varying capabilities in a harmonious fashion is the goal of HSI. This chapter describes some of the HSI process as it relates to risk management, and also introduces some HSI tools and demonstrates their use in applied settings.

25.2
Systems Engineering

A system is defined by the International Council on Systems Engineering (INCOSE) as an artifact created by humans consisting of components that pursue a common

Handbook of Loss Prevention Engineering, First Edition. Edited by J. M. Haight.

goal unattainable by each of the single elements. This definition can lead to very broad generalizations or in-depth plans for an element. The engineering part of SE represents the practice of employing tools and structured approaches to develop a product. Putting these two words together describes the SE practice of defining and documenting requirements for a product or process, preparing or choosing amongst design alternatives, assuring that requirements have been met, and finally deploying, maintaining, and disposing the system. The process is iterative, all the while employing optimization and streamlining the various elements to ensure that cost, schedule, and operational requirements are met.

Forsberg and Cotterman (2000) described the ''Vee'' model relating SE to the project cycle (see Figure 25.1). Design explorations and analyses are conducted at the start of the system development, ending with the complete integration and qualification of the finished system. The left side of the Vee model describes decomposition and definition activities; the center base represents the complete specification of system components, and the right side describes the quantitative verification activities assuring that requirements were met.

As SE practices unfold, problems inherently develop. These challenges are addressed using the systematic approach integral to SE practices and, once sufficiently addressed as defined by the agreed-upon requirements, the process moves on to the next phase and next problem (Software Engineering Institute, 2006). The current standard used in industry and military applications is the EIA-621 (ANSI/EIA, 1995) standard. It applies to the product life-cycle starting from the user needs

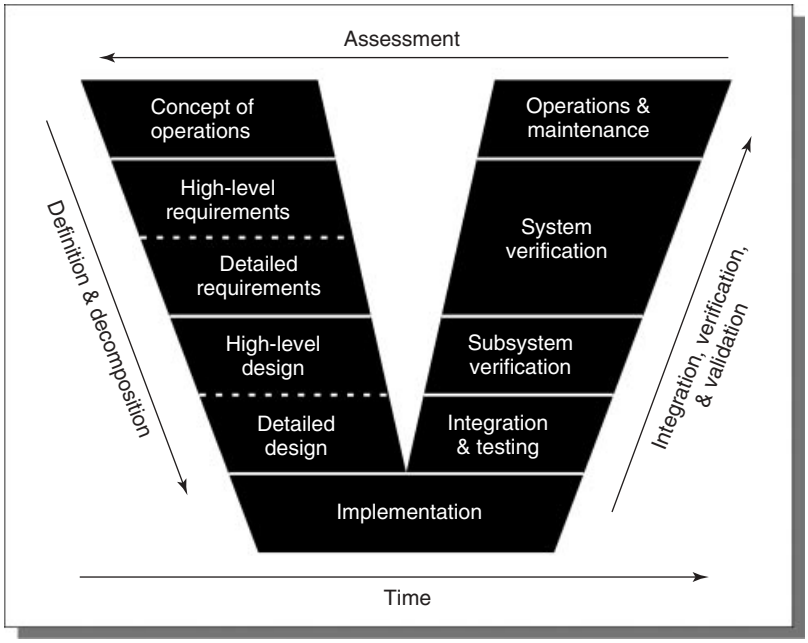

Figure 25.1 Systems engineering Vee model (Osborne *et al*., 2005).

to the final delivery. It outlines 13 related processes divided into five functional groups.

Following SE practices affords a formal design approach that attempts to cover all aspects of design. This includes a robust risk management portion that, given good requirements, will yield a safe design. Safety is not the only benefit of following this approach. Past performance on projects and previous research in SE indicated that there is a positive correlation between utilizing formal SE practices and the degree of success in an engineering undertaking, especially in return on investment (ROI) (Honour, 2006; Ahram and Karwowski, 2009). Today's difficult economy mandates a positive ROI on practically all undertakings and SE practices assure that safety is not compromised.

Profitability or on-time delivery should never take priority over system safety, and having a detailed plan based on SE practices assures that this does not happen. Ensuring a high ROI with a detailed plan on how to execute the design reassures management and moves projects forward (Taubman, 2008).

SE also manages complexity. It is no secret that hardware and software are growing more complex with each passing year. New features are added, additional functionalities are created, and overall system complexity increases accordingly. Baumgart *et al.* (2011) stated that the implementation of new functions can actually lead to ''inconsistent'' systems. Their rationale is that in order to accommodate new functionalities, technical architectures must be modified to compensate for these changes. Having a continuous, robust, reproducible design approach simplified matters greatly. This traceable, repeatable approach can increase overall levels of system safety through explicit visualization of system operations and identify possible breakdowns before they occur.

Another added benefit of a rigorous design framework is enabling concurrent design, where multiple projects proceed in parallel. These methods can also permit reuse of system components in new functions. These components can find new life by re-instantiation. This is especially true of software projects, but many modular physical components also demonstrate this property. SE supports identifying and assessing risk and helps system designers and program managers develop proactive, cost-effective loss prevention programs that protect against loss, safeguard systems, and provide operational continuity. SE assures that a life-critical support system behaves as needed even when components fail.

This merging of system components and functions was not possible until recent times. The design process must evolve to keep up with these new practices. The SE practice comprises the following chain of artifacts: *Processes* → *Methods* → *Tools*. Processes are identified based on previous studies and general design heuristics, or from standards such as the EIA-632 (ANSI/EIA, 1998) standard. Methods can either be developed from scratch or recycled from past programs if they are applicable. The third item, tools, is concerned with methods involved in conceptual and detailed design; the tools are defined, acquired, or created once a viable method exists (Guillerm, Demmou, and Sadou, 2010).

The overall goal of SE is to convert user or stakeholder requirements into technical engineering requirements that drive design. Safety is always an important part

of the requirement process, and is supported by the SE framework through both validation of said requirements and verification that they have been met. Safety requirements generally set constraints on any given system. For example, safety requirements may mandate fall protection provisions, or set touch temperatures on surfaces, or limit shift lengths. Safety requirements are hierarchical in nature, with the most attention and consideration given to those of a critical nature or those needed even when components fail. Stakeholders, regulatory bodies, governing policies, or certification and quality standards may specify safety requirements. These requirements on safety and loss prevention engineering may also be ordered and managed by a set of attributes. Software tools such as Systems Modeling Language (SysML) may aid in this venture (Guillerm, Demmou, and Sadou, 2010).

25.3 Human–Systems Integration

HSI is pertinent to the area of SE. HSI focuses on the interdisciplinary technical and management processes for integrating human capabilities and limitations within and across all system elements. This concept is essential to SE practice. The goal of HSI is to optimize total system performance while accommodating the characteristics of the user population that will operate, maintain, and support the system and minimize system overall life-cycle costs (Folds, Gardner, and Deal, 2008). This process operates throughout system design, development, fielding, sustainment, and retirement. HSI experts work to ensure the proper integration of human capabilities and limitations in systems design. The attention to HSI in system development programs has resulted in hundreds of human-centered design improvements. Efforts were concentrated on maximizing total system performance through improvements in human workload, ease of maintenance, and personnel safety. These efforts resulted in billions of dollars saved and the prevention of hundreds of system-related fatalities and disabling injuries (Booher and Minninger, 2003).

For example, the United States Air Force (USAF) HSI program is designed to support mission critical operations by designing systems that optimize human performance at every level. Two studies conducted by the USAF Science Advisory Board (USAF SAB, 2004, 2005) indicated that the increased need on human operators negatively affected the accuracy of decisions, occupational safety, and total life-cycle cost of the system. This increased workload on human operators' results from the increased volume and complexity of the information, change of job demands, and increased labor constraints resulting in less labor available. Recommended actions included the importance of strengthening HSI methods in the SE processes.

Paul Kaminski, a USAF subject matter expert, indicated the need for SE, engineering management, and incorporating HSI methods by stating the following (Taubman, 2008):

> The central problem is a breakdown in the most basic element of any big military project: accurately assessing at the outset whether the technological goals are attainable and affordable, then managing the engineering to ensure that hardware and software are properly designed, tested and integrated. The technical term for the discipline is systems engineering. Without it, projects can turn into chaotic, costly failures.

The Potomac Institute for Policy Studies stressed the need for developing HSI tools to cover human capabilities and design future systems that are more efficient. Failure to incorporate HSI methodology within the SE process may result in the failure to meet desired system objectives, a poor design, unnecessary burdens on the workers, and in some cases negative environmental impacts that could affect public health and safety. System owners may unnecessarily incur total ownership costs. The long-term success of any system relies heavily upon the effectiveness of its operators, maintainers, supported customers, sustainers, and the support network. A study by the Government Accountability Office (GAO), examining 95 military projects worth $1.6 trillion, reported projected cost overruns totaling $295 billion (40%), and an average delay of 21 months in project schedules (Taubman, 2008). A major cause was the lack of engineering management and HSI guidance. A failure in safety-certified systems is acceptable if, on average, less than one life per 10^9 h of continuous operation is lost to failure; the cost versus loss of lives has been considered appropriate by the Federal Aviation Administration (FAA) at this level (SAE, 2010; Williams, 2011).

In all systems, failure to address long-term, life-cycle issues can result in lost customer confidence, lost market share, product liability, and a decrease in repeat business. HSI experts work within the framework, consisting of processes and methodologies, provided by SE to ensure successful integration of humans and systems. The aforementioned methodologies include the familiar, carefully structured approach to meeting the functional and non-functional requirements. The SE team relies on each branch to assist in analyzing customer requirements. Research has shown (Meilich, 2008) that HSI aspects and components remained, until only recently, lacking any established methodologies or integration tools to link various human aspects to SE models for two reasons: lack of relevant taxonomy linkage to SE needs and poor domain languages. Meilich (2008) found that there are current challenges linking the inconsistent behavior of humans to the predictable behavior of systems and machines. In addition, Meilich (2008) stated that there is a lack of conceptual understanding of the various cognitive aspects that contribute to task effectiveness and overall mission success (Wells *et al.*, 2011). Such lapses in understanding can have drastic safety effects when human capabilities are overestimated.

Context is critical to HSI test and evaluation. For example, the effort and time spent to develop systems use case scenarios during mission task analysis are a good investment, and use case scenarios support identifying and assessing risk and help system designers develop proactive, cost-effective loss prevention actions that protect against loss. Scenarios chosen for HSI demonstration should be those that

are critical for mission success. There are important questions that HSI analysts need to investigate thoroughly:

- Under which conditions are people the most fatigued?
- What are the critical decisions?
- What are their triggers?
- What combination of circumstances generates extreme hazards?

HSI professionals seek to demonstrate the integrated system in situations where performance is critical. The demonstration will evaluate opportunity for human error, keeping in mind that errors can be induced by either equipment failures, the inadequacy of the human–system interface, human actions, or communications errors. Errors can be captured by trained observers, system databases that record system status during tasks or usability tools that track keystrokes and eye movements. Many human error accidents were induced by hardware and software designs that neglected HSI (Ahram, Karwowski, and Amaba, 2010). The HSI evaluation must also demonstrate the survivability and resilience of the integrated system and answer the following:

1) When equipment breaks down, does the system provide insight that enables the operator to execute remedial actions, or does the operations concept call for evacuation?
2) Has the information support plan been modified without HSI review to remove data sets that would enhance the ability of the operator and maintainer to diagnose and respond to anomalies?

Developing HSI test parameters, as shown in Table 25.1, can be an effective strategy for defining systems demonstration. Developing these parameters establishes traceability during mission task analysis (Ahram *et al.*, 2009). Some of the parameters are evaluated against the objective whereas some others require evaluation in the context of use case scenarios in order to be meaningful. Subjective evaluation is required for the more general parameters, which can be done by using rating scales or by administering questionnaires developed by HSI specialists that can be supplemented with interviews. Based on the above discussion, it is clear that HSI is an important aspect of SE that brings human considerations into the system design process and seeks to properly maintain the human elements of the system and assures that a life-critical support system behaves as required.

Human error can be managed and attempts are constantly being made to ''design out'' possibilities for human error. These strategies include, but are not limited to, training, hardware changes, software optimization, and regulation of activities. These interventions have been shown to be effective in increasing safety and reducing error. An example of this is the development of the angled aircraft carrier deck (Wiegmann and Shappell, 2003). This innovation eliminated hazardous opportunities for collisions where an aircraft taking off from the bow aborted the take-off while another was landing. The angled deck changed the direction of take-off aircraft while others could still land safely. An understanding

Table 25.1 Sample human–systems integration test parameters.

Access to amenities	Illumination conditions
Acoustics	Maintenance/installation safety
Atmosphere (temperature, pressure, humidity, quality, etc.)	Maintenance/installation time to complete
Auxiliary equipment and attire form, fit, function	Motivation of performance
Decision correctness	Physiological state as a function of time (fatigue, stress)
Decision, time to make	Range of motion
Disorientation and awareness	Safe, rapid ingress/egress
Effectiveness of error prevention or mitigation designs	Safety restraints
Effectiveness of workspace layout	Sound, vibration effects on performance
Equipment handling by population (weight, force required)	Storage space
Error rate per unit time	Stress levels
Error recovery rate per unit time	Task complexity
Fault identification and correction	Training adequacy
Food and water availability	User population qualification/experience
G-forces or zero-G effectiveness	Waste disposal adequacy
Hazard protections	Variability of human response
Human-system interface effectiveness and usability	Weather conditions
Information transfer	Workload

Adapted from Folds, Gardner, and Deal (2008).

of factors contributing to unsafe conditions, anomalies, or accidents was necessary to manage the undesirable results properly.

Complex automated systems have relegated human operators to more of a supervisory role, intervening only when necessary. This supervisory activity of the human operator has led to the creation of a supervisory control model, consisting of five steps (see Figure 25.2). Sheridan (1992, 2008) identified these control functions and broke them down into their respective human and automation components.

The five functions that humans perform with the aid of automation are as follows:

1) **Plan** – The human must predict and represent the end goal while the automation relays relevant information to completing the task at hand.
2) **Teach** – The human must manipulate controls and symbols to create a representation of the end goal and completion states that the automation ''understands.''
3) **Monitor** – Supervise the execution of the task, subject to intermediate constraints and performance measures; the automation carries out the task.
4) **Intervene** – When current state variables or conditions do not match those prescribed, modify the automation's functioning or disconnect it entirely.

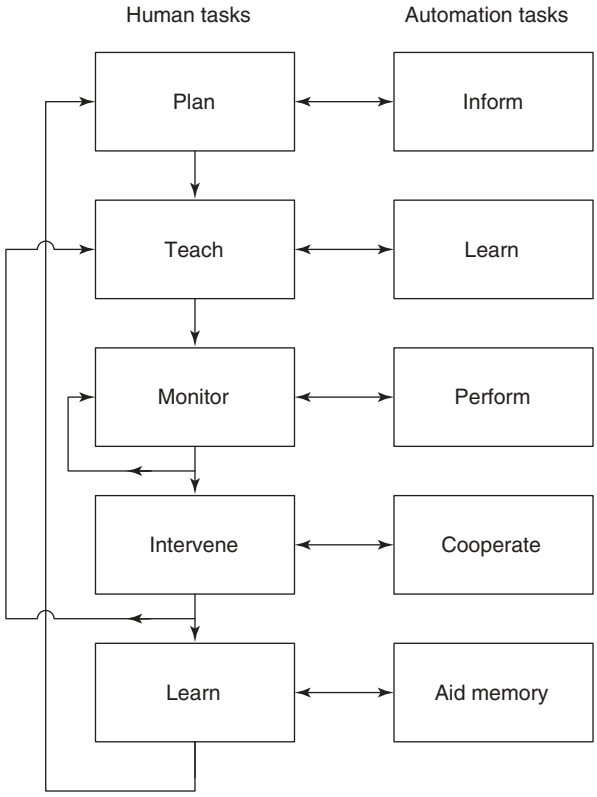

Figure 25.2 Supervisory control framework. (Adapted from Sheridan, (2008).)

5) **Learn** – Develop heuristics and shortcuts, and employ information learned to improve future performance; the automation, with its digital storage capacity, aids in relevant information retrieval.

According to the supervisory control framework, at any point there may be inconsistencies, aberrations, loss of information or complete signal degradation, or outright incorrect information. Understanding such models through reductionist thinking, empirical research, and other visualization methods can aid in identifying possibilities for breakdowns or miscommunications, thereby increasing safety and loss prevention when properly managed or designed out of the system.

Taxonomies have also been employed to aid in organizing information for more effective analyses. Taxonomies serve as maps and guides for a group of related concepts. Individual simplified elements are mapped and plotted, and relationships are created between them to illustrate similarities and dependencies.

Classifying and organizing concepts make them easier to work with, as human learning can be reduced to matching input patterns with output patterns. When the relationships between concepts are easily understood, the pattern matching activity is simplified. For a classification system to be effective, its outputs should have

the same meaning for all users of the system. That is, more usable classification systems are those where higher correlations between elements belonging to the same category reflect actual user perceptions of those items. The main goal of this classification is to provide usable and functional relationships, affording the creation of previously unknown relationships (Fleishman and Quaintance, 1984).

A taxonomy related to errors was created by Swain and Guttman (1983). Their research investigated accidents at nuclear power plants. The categories created were as follows:

- **Errors of omission** – Errors where an individual omits the entire task or omits a step in the task. These errors are failures to perform an action.
- **Errors of commission** – Errors that are due primarily to poor selection. The individual selects a wrong control incorrectly, incorrectly manipulates said control, or issues an incorrect command or input.
- **Errors of sequence** – Errors caused by actions that are performed out of the proper sequential order.
- **Errors of timing** – Errors caused by actions that are either too early or too late.

Classifying error from accident reports and creating data on error counts can provide insights into error causes and, from these data, strategies on error prevention can be created. Furthermore, these taxonomies can be linked directly to organizational processes. Knowing about these error types and their properties can lead to a better understanding of why they occur. Safety and risk management professionals can then use qualitative and quantitative methods to drive changes into existing designs or to provide valuable input into a current design process.

One tool used by SE professionals is SysML. SysML has the attributes necessary to convey system properties meaningfully to multiple parties, be they stakeholders, designers, investigators, or test and validation engineers (OMG, 2011).

25.4 Systems Modeling Language

SysML is an SE tool to aid in managing complexity. At its core, it is a sort of cognitive scaffolding that visually represents complex systems' elements. These representations may be simplified or decomposed, and often contain "pseudocode," a programmer's tool using natural language to serve as a placeholder for actual code. Pseudocode will strongly hint or suggest programming elements. SysML also provides powerful diagrams that clearly define functional system boundaries, providing at-a-glance understanding of where the system begins and ends. These diagrams are especially useful for defining and reminding system designers who or what interacts with the system.

SysML is so very powerful because it integrates, visualizes, and aids in enacting methods and processes from fields related to risk management, namely those from safety engineering, human factors engineering (HFE), and HSI.

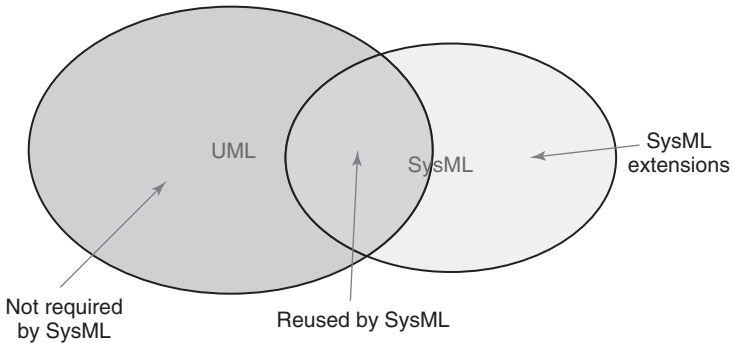

Figure 25.3 Relationship between SysML and UML. (Adapted from OMG, (2011).)

SysML was created and defined by the Object Management Group (OMG). SysML is an extension to the Unified Modeling Language (UML), as defined in Figure 25.3. It builds upon and extends the capabilities of UML, a powerful graphical tool for modeling systems.

The capabilities and benefits that SysML provides to any application managing complexity are far from fully utilized and exploited. This chapter describes the benefits that SysML can bring to risk management, especially from an applied view. The overarching theme of this chapter is that SysML manages complexity, and good management of said complexity offers measurable, discrete, and easily applicable avenues of mitigating risk.

One beneficiary of SysML modeling is safety engineering. According to Thramboulidis and Scholz (2010), system safety is an aspect of any system that enables it to prevent hazards that can cause accidents or losses. The IEC (International Electrotechnical Commission) defines risk of a hazard as

$$\text{risk}_{\text{hazard}} = \left(\text{probability}_{\text{hazard}}\right) \times \left(\text{severity}_{\text{hazard}}\right)$$

The four severity levels are *catastrophic*, *critical*, *marginal*, and *negligible*. Reducing risk according to this definition means reducing the probability of a hazard or the severity of a hazard. Good safety and loss prevention engineering involves defining and predicting what and where possible hazards may occur. Thramboulidis and Scholz (2010) stated that safety analyses are traditionally carried out by safety engineers who have very different training and skillsets to design engineers. This difference in experience introduces a rift that may lead to miscommunication and misunderstanding. There is an analogous situation between HF specialists and design engineering (Chapanis, 1996). Their purported (3 + 1) SysML view model bridges this gap. One powerful element of their process is the employment of use case diagrams to define system/user boundaries succinctly for all members of the product/process design and support community. System architectures are drawn using UML constructs to facilitate understanding. Furthermore, these diagrams can help identify risk possibilities and also ascertain severity levels to said possibilities.

SysML is widely used as a visualization tool to promote complex system understanding to a variety of stakeholders. After all, the end user is not the sole focus

of risk assessment and loss prevention. Expanding the focus of analysis to include stakeholders beyond the end user of a process or product aids design in "designing out" error possibilities and failure modes.

SysML diagram types and their activities are outlined in Figure 25.4. The basic unit of SysML is a "block." Blocks are denoted with **bold** type and encased in guillemets (angle quotes) which are line segments, pointed as if arrows (≪ ≫). These blocks represent hardware. By connecting blocks, hierarchies are quickly formed. These hierarchies are enclosed in rectangles and called "package diagrams."

Johnson *et al.* (2007) described a more behavioral modeling approach using SysML diagrams. They claimed that SysML succinctly defines system structure and behavior employing the following diagram types:

- **Activity diagrams** – Describe inputs, outputs, sequences, and conditions that govern various system behaviors.
- **Sequence diagrams** – Describe the flow of control, commands, and responsibilities between actors and a system or its components.
- **State machine diagrams** – Model discrete behavior and guide developers and designers through finite state transitions in system states.
- **Parametric diagrams** – Model mathematical constraints against system properties.

Guillerm, Demmou, and Sadou (2010) demonstrated that SysML drives requirements integration by assigning various diagrams to visualize the requirements process:

- **The requirements** – Requirements diagram, Use case diagram.
- **The structure** – Block diagram (internal/external).
- **The behavior** – State chart, activity diagram, sequence diagram.
- **The constraints** – Parametric diagram.

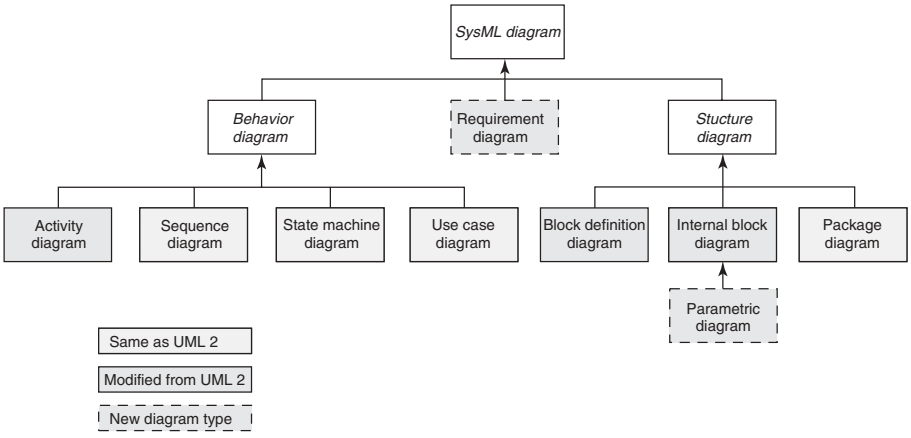

Figure 25.4 SysML diagram types. (Adapted from OMG, (2011).)

Relationships are easily defined and displayed. Requirements can be linked to other components of the model, affording simplified traceability and knowledge of responsibility. Modifications or changes to system properties are easily managed through these diagrams, making impact analysis straightforward.

Nejati *et al.* (2012) developed a framework for software safety inspections using SysML. They cited the need for developing software that is in compliance with safety standards such as IEC 61508 or DO-1788B for airborne systems. Traceability is important, as it affects all areas of development. Software goes through a hazard and risk analysis. This analysis forms the baseline for overall system safety requirements (including both hardware and software). Traceability ensures that links exist from overall system safety requirements to the software requirements. This feature is important for maintenance and development tasks and is necessary for inspections and audits by those doing the certification. Poor traceability is likely to incur omissions, delays, and overall compromises in system safety. This is especially true of complex systems involving long supply chains and many partners in design.

Sommestad, Ekstedt, and Johnson (2010) employed UML diagrams to aid the visualization of system security. They claimed that if security risk could be easily understood from architecture models, then it would be easy to assess risk differences between the ''as is'' configuration with possible ''to be'' alternatives. Such visibility is powerful when making important critical design decisions. The class diagrams employed in the authors' venture encompass internal software security, in addition to external assets such as firewalls, switches, and other data pipelines.

SysML is also employed in reliability studies. A study by David, Idasiak, and Kratz (2010) illustrated how automatic analysis of SysML models is possible, with the output being an FMEA. One challenge found in safety analyses is that they only have the functional behavior of the system to work with. They address this challenge with their SysML models and diagrams which utilize the following three steps:

1) deduction of the dysfunctional behaviors with an FMEA along with identification of the impacted requirements
2) model construction integrating functional and dysfunctional behaviors employing formal language
3) analysis and quantification of dysfunctional behavior.

These iterative steps can be reused according to approaches outlined in various safety standards. Their powerful approach also contains a set of operators employed by an algorithm to construct viable FMEA from SysML diagrams. This automated approach is thorough because during creation of the SysML diagrams the system must be fully decomposed, leaving little room for omission of safety issues. Furthermore, owing to the comprehensive analysis of the system prior to creating the FMEA, requirements are much less likely to be overlooked.

One challenge in any risk management environment is that of handling distributed cognition. Distributed cognition refers to the dynamics that result when

groups of people interact with systems, interfaces, and each other to complete a task or accomplish a goal. The mantra in handling distributed cognition is that there is no one good approach: each case is unique because of the inherent variability of humans, their interactions, and the many combinations and permutations of these interactions. Ethnography is traditionally used to investigate usability, interface design, and safety and to gather requirements in such scenarios (Hollan, Hutchins, and Kirsh, 2000).

SysML can be used to assist with the requirements analysis portion of the SE process, along with a traceability matrix that links requirements to system properties. The visual impact of SysML and its easy-to-understand diagrams bind system and user responsibilities and support validating safety requirements. Upon validation, the diagrams serve as reminders for verification activities. SysML can aid in system simulation activities to verify safety requirements. Discrete simulation and event modeling provide a powerful verification tool. Wang and Dagli (2008) employed SysML-based specifications to drive Colored Petri Nets (CPNs), thus permitting a robust dynamic and static analysis of system safety properties. Petri Nets are a type of mathematical modeling language focused on system transitions. They provide graphical representations of stepwise processes. By using executable models coded from SysML, requirements verification and safety studies become much simpler. Where executable models would be too time consuming to create, CPNs offer a fast alternative to visualizing system properties for verification activities. Once again the benefits offered by SysML are a more complete understanding of the system prior to implementation, traceability for requirements, visualization of complex system functions, speed, and thoroughness of system safety analyses earlier in the design process when changes can still be made.

One area of product design that has demonstrated shortcomings in safety analysis according to the literature is that of embedded system design (Kaiser *et al.*, 2010). The authors claimed that hazard analysis and creation of technical safety concepts are not tightly integrated with core design activities. This decoupling results in a lack of traceability and consistency. The authors presented a comprehensive approach to mediating these shortcomings using current techniques employed in SE and safety science. SysML and UML were employed to construct a consistent safety framework early in design. The goal was to begin hierarchical system modeling and feature allocation early in the design process. Traditional SE software, IBM Rational DOORS (Dynamic Object Oriented Requirements System), was employed in its usual fashion: keeping track of requirements, their dependencies, and relations, and providing traceability for the entire process. UML diagrams were employed for simpler system function such as discrete environmental interactions (switches). For continuous applications, SysML provided advanced modeling stereotypes. SysML block definition diagrams were then easily converted to FMEAs by providing the underlying structure necessary for this task. Findings from this study included the fact that experts involved in the research felt they had a better understanding with increased agreement even by using the informal block diagrams representing the system drawn up in Microsoft Visio.

One often touted benefit of SysML through this work is that it offers traceability to requirements. Tracing requirements back to system functions is integral to safety and hazard analyses. Briand *et al.* (2010) designed an experiment to test if SysML design slices could support safety investigations. Slicing is a technique developed to help manage complex software systems, which have been growing exponentially in recent years. Slicing is traditionally used to aid in debugging software as only sections are looked at one a time, rather than the entire program. Slicing is applicable to safety and loss prevention engineers and investigators as it narrows the investigative scope to only relevant elements regarding safety and risk management. The software slices that were investigated maintained their traceability links. By employing appropriate slicing activities and pairing them with powerful SysML visualization techniques, the authors were able to demonstrate practical significant benefits to the correctness of inspections, better compliance decisions, and reduced effort in safety investigations.

UML has also made forays in SE, benefiting the process. Willard (2007) described ''straightforward benefits'' of SysML and also some of its challenges. The most lauded benefit of employing UML and, by extension, SysML to SE, safety science, loss prevention engineering, and risk management is that it provides a standard and comprehensive system specification paradigm. The symbols, diagrams, and relationships used in UML are not ambiguous, and thus provide a stable communication medium across engineering and safety disciplines. This communication is especially important when describing system structure versus system behavior. UML diagrams have descriptive depictions that clearly lay out structures and behaviors.

Figure 25.5 depicts the SysML diagram taxonomy with HSI model diagram considerations. The HSI diagram components have been identified under both the behavior and structure diagrams. This has been done to compensate for a human capabilities diagram and task requirements with respect to human performance, which does not exist. Figure 25.6 provides a summary of the system model with

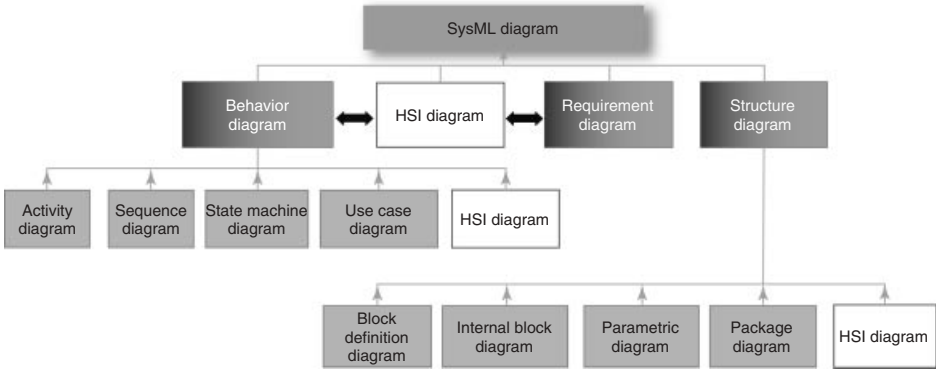

Figure 25.5 Extended SysML diagram taxonomy incorporated with HSI considerations (extended model based on original framework by Friedenthal, Moore, and Steiner, 2008).

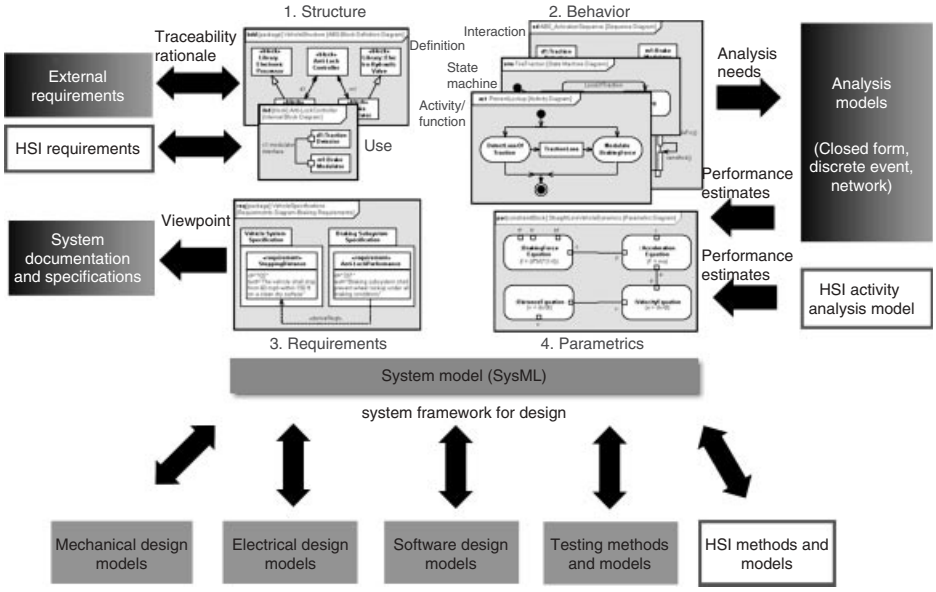

Figure 25.6 Extended system model incorporating HSI as a framework for analysis and traceability (extended model based on original framework by Friedenthal, Moore, and Steiner. 2008).

HSI as a framework for analysis and traceability. Both diagrams have been extended here from originals by Friedenthal, Moore, and Steiner (2008).

In model-based systems engineering (MBSE), human behavior model libraries and human behavior analysis can be integrated with user domain model libraries. Friedenthal, Moore, and Steiner (2008) stated that ''MBSE is the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life-cycle phases.'' Model-based HSI can add value to SE by the application of Jazz technology and MBSE methodologies (Ahram *et al.*, 2009; Karwowski and Ahram, 2009). Jazz is an open, scalable, extensible team collaboration technology developed as a joint project between IBM Rational and IBM Research for seamlessly integrating work across the development (both systems development and software development) life-cycle. Jazz technology allows collaborative software delivery, redefines software development, permits comprehensive safety systems analysis and human integration aspects, and fosters successful HSI modeling by leveraging human performance needs and tasks requirements into systems design and modeling, thereby adopting MBSE techniques early in the process.

25.5
Human–Systems Integration Model Domains

The human domains identified internationally and shown in Figure 25.7 include HFE, manpower, personnel, training, safety and health hazards, habitability and survivability, and are foundational human-centered domains which are discussed below.

1) **Manpower** – Manpower addresses the number and type of personnel in the various occupational specialties required, and potentially available, to train, operate, maintain, and support the deployed system, based on work and workload analyses.
2) **Personnel** – Personnel considers the type of human knowledge, skills, abilities, experience levels, and human aptitudes (i.e., cognitive, physical, and sensory capabilities) required to operate, maintain, and support a system and the means to provide (recruit and retain) such people.
3) **HFE** – HFE involves understanding and comprehensive integration of human capabilities (cognitive, physical, sensory, and team dynamic) into a system design, beginning with conceptualization and continuing through system disposal.
4) **Environment** – Environment considers conditions in and around the system that affect the human's ability to function as part of the system, and the measures necessary to protect the total system from the environment and to protect the environment and all living things and systems from the systems design, development, manufacturing, operations, sustainment, and disposal activities.

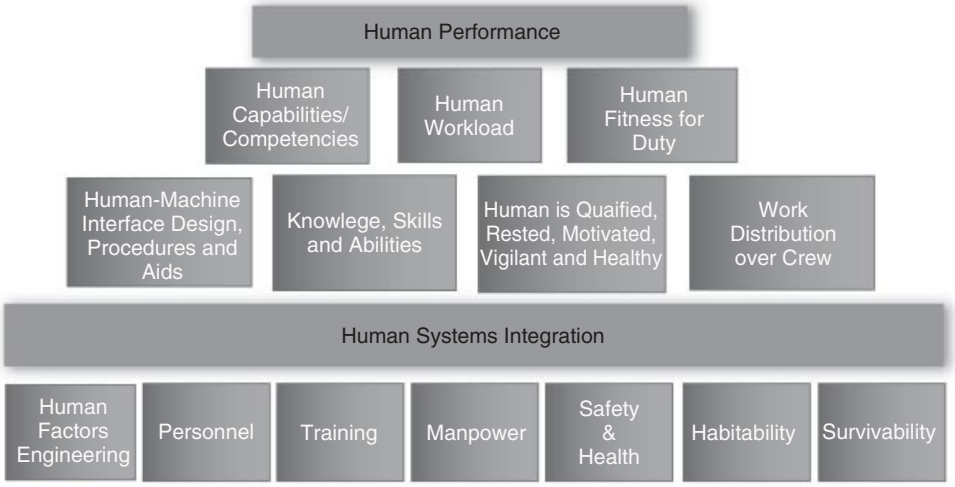

Figure 25.7 Domains of human systems integration. (Adapted from USAF SAB, (2004).)

5) **Safety and occupational health** – Safety promotes system design characteristics and procedures that minimize the potential for accidents or mishaps that cause death or injury to operators, maintainers, and support personnel, that threaten the operation of the system, or that cause cascading failures in other systems. Occupational health promotes system design features and procedures that minimize the risk of injury, acute or chronic illness, or disability, and enhance job performance of personnel who operate, maintain, or support the system.
6) **Habitability** – Habitability involves characteristics of system living and working conditions such as lighting, ventilation, adequate space, vibration, noise, temperature control, availability of medical care, food and drink services, suitable sleeping quarters, sanitation, and personnel hygiene facilities.
7) **Survivability** – Survivability addresses the characteristics of a system (e.g., life support, personal protective equipment, shielding, egress or ejection equipment, air bags, seat belts, electronic shielding) that reduce susceptibility of the total system to operational degradation or termination, to injury or loss of life, and to a partial or complete loss of the system or any of its components.

SysML has greatly benefited traditional software and hardware design. Its tightly defined elements that offer no ambiguity in their meaning provide, at the very least, a powerful visualization tool for safety investigators and loss prevention engineers. These visualizations can be taken back to design or management to drive positive changes affecting safety into the system. The most variable component in these systems is the human element, serving as saviors of a system and not just a loose, uncontrolled element.

Safety and loss prevention engineering are not the only beneficiaries of this approach; following SE practices and methodologies affords a formal and embedded loss prevention design approach that attempts to cover all aspects of safe and reliable design, where the human element is the core of all systems. This includes a robust risk management portion that, given good requirements, will yield a safe design. Furthermore, traceability analysis can aid in redefining requirements or identifying corroborating factors should an unfortunate incident occur. The application of SysML in the context of systems, safety, and loss prevention engineering, will benefit the design and safe operation of complex technological systems in the future.

References

Ahram, T.Z., Karwowski, W., and Amaba, B. (2010) User-centered systems engineering approach to design and modeling of smarter products. In 5th IEEE International Conference on System of Systems Engineering (SoSE), Loughborough University, June 22–24, 2010, pp. 1–6.

Ahram, T.Z., Karwowski, W., Amaba, B., and Obeid, P. (2009) Human systems integration: development based on sysML and the rational systems platform. In Proceedings of the 2009 Industrial Engineering Research Conference, Miami, FL, pp. 2333–2338.

Ahram, T.Z., Karwowski, W. (2009) Human Systems Integration Modeling. Human Factors and Ergonomics Society Annual Meeting Proceedings, **53**, 24, pp. 1849–1853 (5). 53rd Annual Meeting of the Human Factors and Ergonomics

Society (HFES 2009) , October 19–23, 2009, San Antonio, Texas, USA

ANSI/EIA (American National Standards Institute/Electronic Industries Alliance) (1995) 631. *Consumer Electronics Group Product and Packaging Bar Code Standard*.

ANSI/EIA (American National Standards Institute/Electronic Industries Alliance) (1998) 632. *Processes for Engineering a System, http://www.davi.ws/avionics/TheAvionicsHandbook_Cap_24.pdf* (last accessed 23 January 2012).

Baumgart, A., Reinkemeier, P., Rettberg, A., Stierand, I., Thaden, E., and Weber, R. (2011) A model-based design methodology with contracts to enhance the development process of safety–critical systems, in *Software Technologies for Embedded and Ubiquitous Systems* (eds S.L. Min, R. Pettit, P. Puschner, and T. Ungerer), Springer, Berlin, pp. 59–70.

Booher, H.R. and Minninger, J. (2003) Human systems integration in army systems acquisition, in *Handbook of Human Systems Integration* (ed. H. Booher), John Wiley & Sons, Inc., New York, pp. 663–698.

Briand, L., Falessi, D., Nejati, S., Sabetzadeh, M., and Yue, T. (2010) Traceability and SysML Design Slices to Support Safety Inspections: a Controlled Experiment, Simula Research Laboratory Technical Report 2010-08, Simula Research Laboratory, Lysaker, Norway.

Chapanis, A. (1996) *Human Factors in Systems Engineering*, Wiley Series in Systems Engineering and Management, John Wiley & Sons, Inc., New York.

David, P., Idasiak, V., and Kratz, F. (2010) *Reliab. Eng. Syst. Saf.*, **95** (4), 431–450.

Fleishman, E.A. and Quaintance, M.K. (1984) *Taxonomies of Human Performance*, Academic Press, San Diego, CA.

Folds, D., Gardner, D., and Deal, S. (2008) *INCOSE INSIGHT*, **11** (2), 15–18.

Forsberg, K., Mooz, H., and Cotterman, H. (2000) *Visualizing Project Management: a Model for Business and Technical Success*, John Wiley & Sons, Inc., New York.

Friedenthal, S., Moore, A., and Steiner, R. (2008) *A Practical Guide to SysML: the Systems Modeling Language*, Morgan Kaufmann, Burlington, MA.

Guillerm, R., Demmou, H., and Sadou, N. (2010) Information model for model driven safety requirements management of complex systems, in *Complex Systems Design and Management*, (eds M. Aiguier, F. Bretaudeau, and D. Krob), Springer, Berlin, pp. 99–111.

Hollan, J., Hutchins, G., and Kirch, D. (2000) Distributed cognition: toward a new foundation for HCI research, *TOCHI*, **7** (2), 174–196.

Honour, E.C. (2006) A practical program of research to measure systems engineering return on investment (SE-ROI). Presented at the 16th Annual Symposium of the International Council on Systems Engineering, Orlando, FL.

Johnson, T.A., Paredis, C., Burkhart, R., and Jobe, J. (2007) Modeling continuous system dynamics in SysML. Presented at the ASME International Mechanical Engineering Congress and Exposition, Seattle, WA.

Kaiser, B., Klaas, V., Schulz, S., Herbst, C., and Lascych, P. (2010) Integrating system modelling with safety activities, in *Computer Safety, Reliability, and Security* (ed. E. Schoitsch), Springer, Berlin, pp. 452–465.

Karwowski, W. and Ahram, T.Z. (2009) *J. Inform. Syst. Manage.*, **26** (3), 262–274.

Meilich, A. (2008) INCOSE MBSE Initiative Status of HSI/MBSE Activity.

Nejati, S., Sabetzadeh, M., Falessi, D., Briand, L., and Coq, T. (2012) A SysML-based approach to traceability management and design slicing in support of safety certification: framework, tool support, and case studies. *J. Inform. Software Technol.*, **54** (6), 569–590.

OMG (2011) OMG SysML – OMG Systems Modeling Language, *http://www.omgsysml.org/* (last accessed 26 October 2011).

Osborne, L., Brummond, J., Hart, R., Zarean, M., and Conger, S. (2005) *Clarus Concept of Operations*, Publication No. FHWA-JPO-05-072, Federal Highway Administration (FHWA), Washington, DC.

SAE (2010) ARP4754A *Guidelines for Development of Civil Aircraft and Systems*, Society of Automotive Engineers,

http://standards.sae.org/arp4754a (last accessed 22 June 2012).

Sheridan, T.B. (1992) *Telerobotics, Automation and Human Supervisory Control*, MIT Press, Cambridge, MA.

Sheridan, T.B. (2008) *J. Hum. Factors Ergon. Soc.*, **50**, 418–426.

Software Engineering Institute (2006) Ultra-Large-Scale Systems: the Software Challenge of the Future, Software Engineering Institute, Carnegie Mellon University, *http://www.sei.cmu.edu/uls/* (last accessed September 6, 2012).

Sommestad, T., Ekstedt, M., and Johnson, P. (2010) *Comput. Secur.*, **29** (6), 659–679.

Swain, A. and Guttman, H. (1983) *Handbook of Human Reliability Analysis with Emphasis on Nuclear Power Plant Applications*, NUREG/CR-1278, Nuclear Regulatory Commission, Washington, DC.

Taubman, P. (2008) Top engineers shun military; concern grows. *The New York Times*, 25 June; correction, 26 June, *http://www.nytimes.com/2008/06/25/us/25engineer.html* (last accessed 22 June 2012).

Thramboulidis, K. and Scholz, S. (2010) Integrating the 3 + 1 SysML view model with safety engineering. In Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on Emerging Technology and Factory Automation, pp. 1–8.

USAF SAB (2004) Human–System Integration in Air Force Weapon Systems Development and Acquisition. SAB-TR-04-04. US Air Force Science Advisory Board, Washington, DC.

USAF SAB (2005) System-of-Systems Engineering for Air Force Capability Development. SAB-TR-05-04. US Air Force Science Advisory Board, Washington, DC.

Wang, R. and Dagli, C.H. (2008) An executable system architecture approach to discrete events system modeling using SysML in conjunction with Colored Petri Nets. In 2nd Annual IEEE Systems Conference 2008, pp. 1–8.

Wells, W., Karwowski, W., Sala-Diakanda, S., Williams, K., Ahram, T., and Pharmer, J. (2011) *J. Univers. Comput. Sci.*, **17** (9), 1261–1280.

Wiegmann, D. and Shappell, S. (2003) *A Human Error Approach to Aviation Accident Analysis: the Human Factors Analysis and Classification System*, Ashgate Publishing, Aldershot.

Willard, B. (2007) *Comput. Stand. Interfac.*, **29** (1), 69–81.

Williams, J. (2011) We're on a mission: taking the mystery out of temporary flight restrictions, *FAA Safety Briefing*, (May/June) 16–18. *http://www.faa.gov/news/safety_briefing/2011/media/MayJun2011.pdf* (last accessed 2 February 2012).