

# Life-Cycle Models for Survivable Systems

*Nancy R. Mead, Robert Ellison, Richard C. Linger, Howard F. Lipson, John McHugh*  
CERT<sup>®</sup>/Software Engineering Institute  
Carnegie Mellon University

Copyright © 2000 IEEE. Published in the Proceedings of the Third Information Survivability Workshop (ISW-2000), October 24-26, 2000, Boston MA, USA\*

## 1 SURVIVABILITY AND THE SYSTEM LIFE CYCLE

Today's large-scale, highly distributed, networked systems improve the efficiency and effectiveness of organizations by permitting whole new levels of organizational integration. However, such integration is accompanied by elevated risks of intrusion and compromise. Incorporating survivability capabilities into an organization's systems can mitigate these risks. As an emerging discipline, survivability builds on related fields of study (e.g., security, fault tolerance, safety, reliability, reuse, performance, verification, and testing) and introduces new concepts and principles. Survivability focuses on preserving essential services, even when systems are penetrated and compromised [Anderson 97, Ellison 99].

Current software development life-cycle models are not focused on creating survivable systems, and often exhibit shortcomings in developing systems with a high degree of assurance of survivability [Marmor-Squires 88]. If addressed at all, survivability is often relegated to a separate thread of project activity, and treated as an add-on property. This isolation of survivability considerations from primary system development tasks results in an unfortunate separation of concerns. Survivability should be integrated and treated on a par with other system properties, to develop systems with required functionality and performance that can also withstand failures and compromises. Important design decisions and tradeoffs become more difficult when survivability is not integrated into the primary development life cycle. Separate threads of activities are expensive and labor-intensive, often resulting in duplication of effort in design and documentation. In addition, tools for supporting survivability engineering are often not integrated into the software development environment. With separate threads of activities, it becomes more difficult to adequately address high-risk issues of survivability and consequences of failure. In addition, technologies that support survivability goals, such as formal specification, architecture tradeoff methods, intrusion analysis, and survivability design patterns, are not effectively applied into the development process.

For each life-cycle activity, survivability goals should be addressed and methods to improve survivability incorporated. In some cases, existing development methods can enhance survivability. Current research is creating new methods that can be applied; however, more research and experimentation is required before the goal of survivability can become a reality.

---

<sup>®</sup> CERT and CERT Coordination Center are registered in the U.S. Patent and Trademark Office.

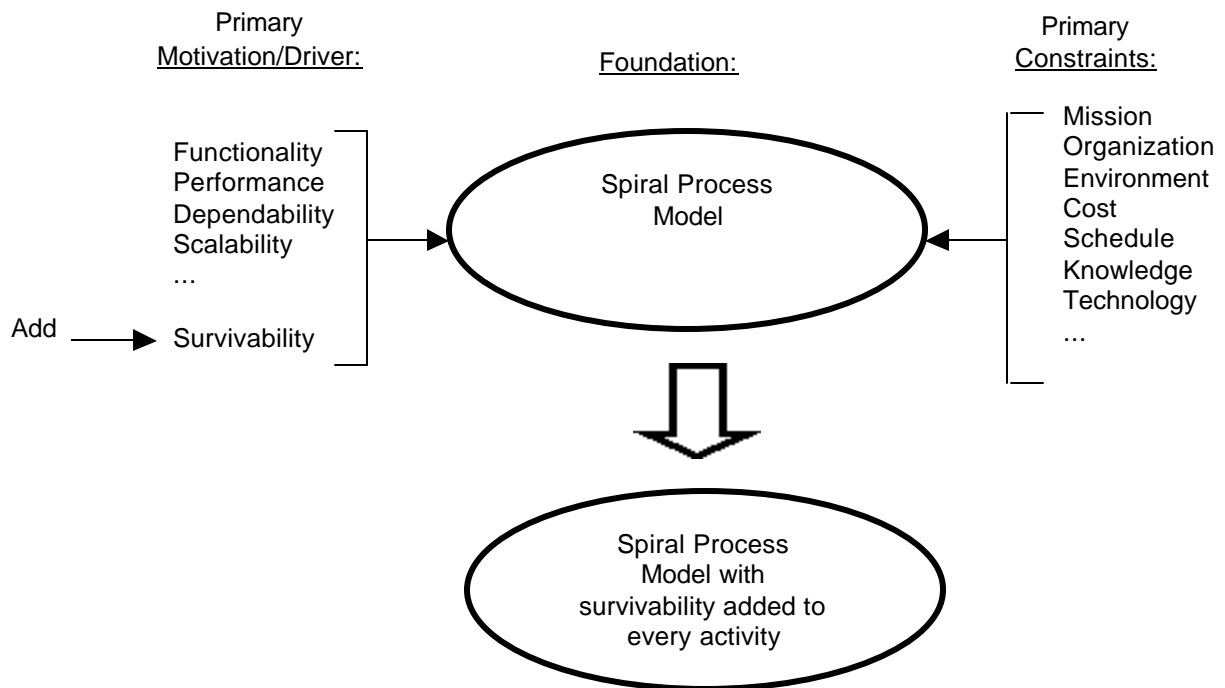
\* Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE. Contact: Manager, Copyrights and Permissions/IEEE Service Center/445 Hoes Lane/P.O. Box 1331/Piscataway, NJ 08855-1331, USA. Telephone: +Intl. 732-562-3966.

## 2 SYSTEM DEVELOPMENT LIFE-CYCLE MODELS FOR SURVIVABILITY

Most development organizations find it necessary to use different life-cycle models for different projects, depending on the nature of the project. Some of the approaches in use today include the waterfall model, spiral model, COTS integration model, incremental development model, and evolutionary model, to name just a few. In this paper, we select one model, the spiral model, and adapt it for survivable systems. We also discuss typical development activities, and illustrate how survivability might be incorporated into each activity.

The generalized “pure” spiral process [Mills 86, Boehm 89] provides a framework for more specialized models. Specialization and enhancement consist of adapting the activities carried out under the model to the special requirements of the systems to be produced. This is done by specifying (1) activities that address the drivers that characterize the system, and (2) constraints that characterize the environment in which the system is to be produced.

The primary driver in the present context is the requirement to develop survivable systems. Constraints include the political and social environments in which the system is to be constructed, the ever-present cost considerations, and the limitations of technologies and knowledge that can be brought to bear on the problem. These combine to yield a specialized version of the spiral model that integrates survivability into the management process, as depicted in Figure 1 [Marmor-Squires 89].



**Figure 1. Specialization of the Spiral Model for Survivability Driver**

Survivable systems must satisfy a variety of conflicting interests. End users want these systems to carry out their primary operational mission, possibly at the expense of violating security policies under some circumstances. It is often the case that systems must also satisfy some certification or accreditation authority. The steps required for these approvals may conflict with the interests of users. And developers want to finish the job, preferably ahead of schedule and under budget. Within the development organization, tensions may exist between the various specialties involved. Resolving these conflicts may involve constraining the environment and the development process. In addition, cost considerations are always present. The spiral development process has proven to be more cost effective than traditional methods, but exhibits a unique distribution of cost with time. Under the spiral model, expenditures are typically higher in early specification and design activities, resulting in cost savings in later implementation and integration activities.

Table 1 identifies a typical set of broad system development activities and the corresponding survivability elements of each. The key point is that survivability is integrated into the broader activities. For example, in defining system requirements, function, performance, dependability, scalability, and other properties must be defined, as well as survivability attributes. The activities in Table 1 comprise the survivability subject matter for project management under the specialized spiral model of Figure 1.

In illustration, consider the following imagined application of the spiral management process to the architecture definition phase. Assume that prior phases have been completed successfully and that the appropriate requirements and specification documents are at hand. The task of the initial architecture definition spiral is to define a set of candidate components and their interconnections that will implement the specified services in a way that satisfies both functional and non-functional requirements. The architect will choose candidate platforms, allocate functions to them, and determine the appropriate connections among platforms and between platforms and the outside world. A variety of tools and techniques will be used to analyze the proposed architecture to determine whether it satisfies the requirements and specifications. One possibility of this analysis is that the proposed architecture satisfies the functional requirements but cannot achieve required throughput. Although processor replication has been used to improve performance, the processors appear to require greater than anticipated coupling for synchronization, and their co-location creates a vulnerability as a potential single site of failure that impacts survivability. Another spiral over the architecture is required to resolve these risks.

Examination of the specification for the service that causes the performance bottleneck reveals that what appeared to be a monolithic service actually decomposes in a way that reduces the processing load and allows the two parts of the service to be separated both physically and temporally. After confirming that this revised service specification satisfies the requirements and is consistent with the other, unchanged specifications, the architecture is revisited. The revised specification permits a reduction in processor load and allows the critical service to be performed at several separate locations with greatly relaxed data-synchronization requirements. As a result, it is possible to configure the system with sufficient redundancy so that at least two loss-of-site events can be tolerated without loss of the service. Further site loss will reduce service levels, but it is possible to prioritize requests so that the minimum essential service level will be maintained. Detailed analyses of this approach shows a low probability race condition that could deadlock the system. Adding explicit synchronization mechanisms (another iteration) and additional communications capacity reduces the residual survivability risk to an acceptable level, and the architecture phase is complete after two spirals of the management process.

<b>Life-Cycle Activities</b>	<b>Key Survivability Elements</b>	<b>Examples</b>
Mission definition	Analysis of mission criticality and consequences of failure	Estimation of cost impact of denial-of-service attacks
Concept of operations	Definition of system capabilities in adverse environments	Enumeration of critical mission functions that must withstand attacks
Project planning	Integration of survivability into life-cycle activities	Identification of defensive coding techniques for implementation
Requirements definition	Definition of survivability requirements from mission perspective	Definition of access requirements for critical system assets during attacks
System specification	Specification of essential service and intrusion scenarios	Definition of steps that compose critical system transactions
System architecture	Integration of survivability strategies into architecture definition	Creation of network facilities for replication of critical data assets
System design	Development and verification of survivability strategies	Correctness verification of data-encryption algorithms
System implementation	Application of survivability coding and implementation techniques	Definition of methods to avoid buffer overflow vulnerabilities
System testing	Treatment of intruders as users in testing and certification	Addition of intrusion usage to usage models for statistical testing
System evolution	Improvement of survivability to prevent degradation over time	Redefinition of architecture in response to changing threat environment

**Table 1. Life -Cycle Activities and Corresponding Survivability Elements**

### **3. Future Research Plans**

Our plans are to study the life-cycle models mentioned earlier in this paper and to determine how survivability considerations can best be incorporated into some of these models. It is also possible that new life-cycle models for survivable systems may emerge. We welcome feedback on the work to date and advice on future research directions.

## References

- [Anderson 97] R. H. Anderson, A. C. Hearn, and R. O. Hundley, "RAND Studies of Cyberspace Security Issues and the Concept of a U.S. Minimum Essential Information Infrastructure," *Proceedings of the 1997 Information Survivability Workshop*, CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University, 1997.
- [Boehm 89] B. W. Boehm, *Software Risk Management*, IEEE Computer Society Press, 1989.
- [Ellison 99] R. Ellison, D. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff, and N. R. Mead, *Survivable Network Systems: An Emerging Discipline*, CMU/SEI-97-TR-013, November 1997, revised May 1999.
- [Marmor-Squires 88] A.B. Marmor-Squires and P.A. Rougeau, "Issues in Process Models and Integrated Environments for Trusted Systems Development," *Proceedings of the 11th National Computer Security Conference*. October 1988.
- [Marmor-Squires 89] Ann Marmor-Squires, John McHugh, Martha Branstad, Bonnie Danner, Lou Nagy, Pat Rougeau, and Dan Sterne, "A Risk Driven Process Model for the Development of Trusted Systems," *Proceedings of the 1989 Computer Security Applications Conference*. Tucson, December 1989. PP 184-192.
- [Mills 86] H. D. Mills, R. C. Linger, and A. R. Hevner, *Principles of Information Systems Analysis and Design*, Academic Press, New York, NY, 1986.