National Aeronautics and Space Administration

Architecture for Integrated Medical Model Dynamic Probabilistic Risk Assessment

D. A. Jaworske[1], J. G. Myers[1], D. Goodenow[1], M. Young[2], and J. D. Arellano[3]
[1]NASA Glenn Research Center, Cleveland, OH, [2]Wyle Science, Technology and Engineering Group, Houston, TX, and [3]MEI Technologies, Houston, TX.

Abstract: Dynamic Probabilistic Risk Assessment (DPRA) predicts complex system outcomes based on many initiating event probabilities and a progression of dependencies.
The Integrated Medical Model (IMM) predicts astronaut health as governed by medical event probabilities and treatment resources. ***The next generation software architecture will use DPRA to merge the IMM with other spacecraft probabilistic models***, based on four significant requirements.

The four Significant Requirements are:

1) To implement ***discrete dependent functions*** – a progression of dependencies
1) To enable ***a diagnosis and treatment capability continuum*** – where the continuum informs both resource utilization and success rate
2) To implemen**t a group exposure event** – infectious disease, a solar particle event, or micrometeoroid impact
3) To create **inputs & outputs to other PRAs** by incorporating "hooks" to the outside world – responding to a CO_2 scrubber failure or activation of the fire control module

Over many Monte Carlo generated instances, ***Evacuation & Loss of Crew Life*** outcomes are binned and ***Crew Health Index*** is calculated based on simulated time lost due to medical events.[2]

Objective: To provide a systematic means of ***creating, documenting, and communicating*** the Integrated Medical Model Dynamic Probabilistic Risk Assessment software design.

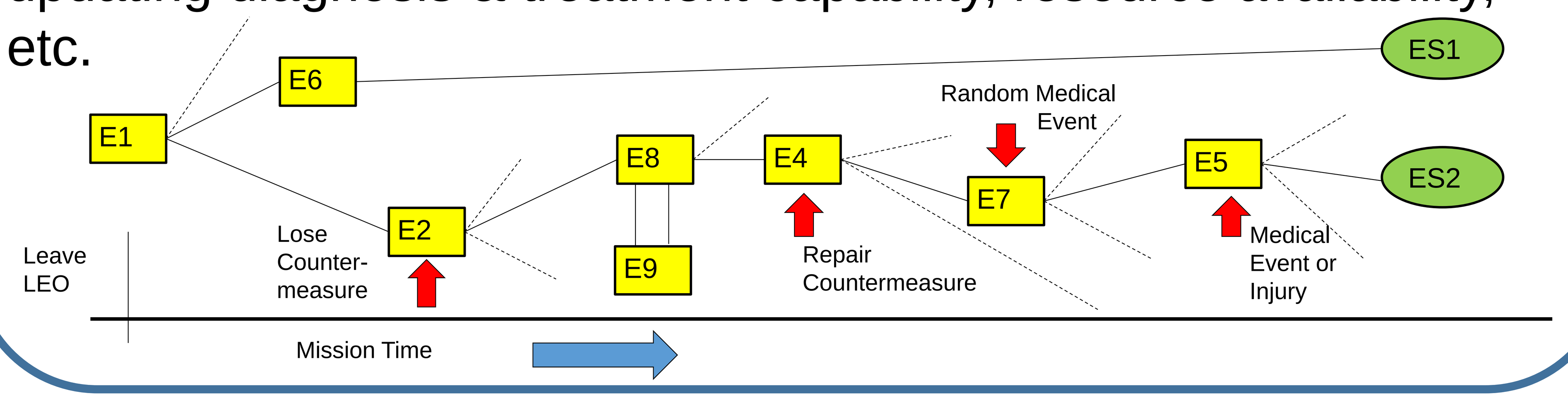
There are ***six key steps*** to managing the software architecture process:[1]

1) Understanding the significant requirements
2) Creating the architecture
3) Documenting and communicating the architecture
4) Evaluating the architecture
5) Implementing & testing the system based on the architecture
6) Ensuring architectural conformance

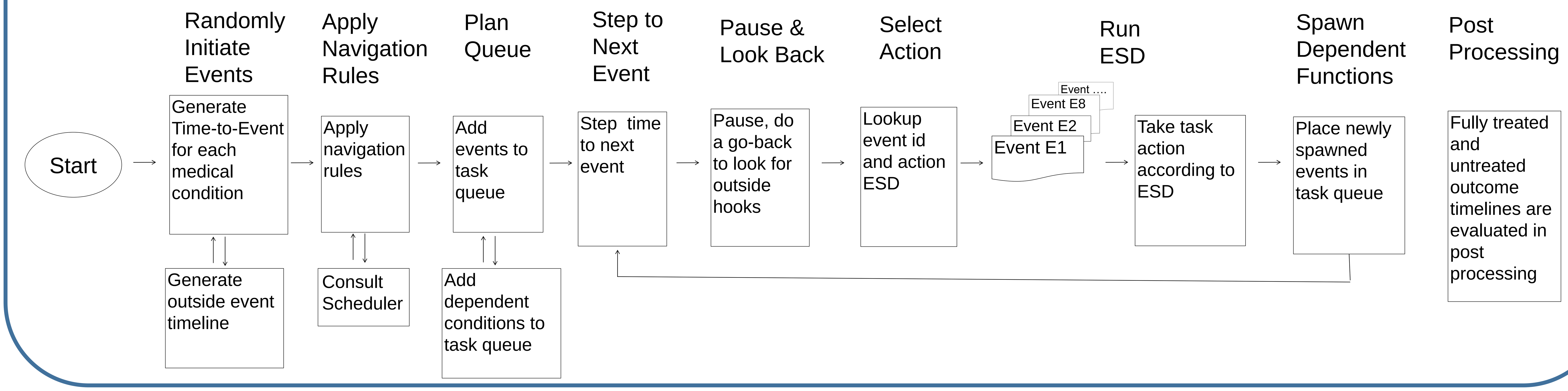
Like project management, the architecture process starts with establishing significant requirements

Planner creates map – Scheduler navigates through map[3]

A planner creates the task queue with discrete dependent functions, look backs, invokes ESDs, & creates hooks to the outside. The scheduler manages the navigation process – updating diagnosis & treatment capability, resource availability, etc.



The architecture selected to meet the four significant requirements is a ***task queue timeline***, to queue up Event Sequence Diagrams (ESDs), spawn dependent functions, update diagnosis capability, move forward in time, pause to look back in time for group events & hooks to the outside, and purposely send hooks to the outside.



[1]Bass, L., Clements, P., and Kazman, R., *Software Architecture in Practice, 3rd Edition*, Addison-Wesley, Upper Saddle River, NJ, 2013.
[2]Keenan, A., Young, M., Saile, L., Boley, L., Walton, M., Kerstman, E., Shaw, R., Goodenow, D. A., and Myers, Jr., J. G. (2015) 45th International Conference on Environmental Systems, Bellevue, WA, ICES-2015-71.
[3]Hu, Y. (2005) Dissertation submitted to the University of Maryland, College Park, MD, 90-124.