Architecture for Integrated Medical Model Dynamic Probabilistic Risk Assessment

D. A. Jaworske¹, J. G. Myers¹, D. Goodenow¹, M. Young², and J. D. Arellano³

¹NASA Glenn Research Center, Cleveland, OH, ²Wyle Science, Technology and Engineering Group, Houston, TX, and ³MEI 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.

**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) 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

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.

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.

The four Significant Requirements are:
1) To implement **discrete dependent functions** – a progression of dependencies
2) To enable **a diagnosis and treatment capability continuum** – where the continuum informs both resource utilization and success rate
3) To implement **a group exposure event** – infectious disease, a solar particle event, or micrometeoroid impact
4) To create **inputs & outputs to other PRAs** by incorporating "hooks" to the outside world – responding to a CO₂ scrubber failure or activation of the fire control module

Planer creates map – Scheduler navigates through map

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.

**Start**

- Randomly Initiate Events
  - Generate Time-to-Event for each medical condition
  - Generate outside event timeline

- Apply Navigation Rules
  - Apply navigation rules
  - Consult Scheduler

- Plan Queue
  - Add events to task queue
  - Add dependent conditions to task queue

- Step to Next Event
  - Step time to next event
  - Pause, do a go-back to look for outside hooks

- Pause & Look Back
  - Lookup event id and action ESD
  - Take task action according to ESD

- Select Action
  - Place newly spawned events in task queue

- Run ESD
  - Fully treated and untreated outcome timelines are evaluated in post processing

- Spawn Dependent Functions