Request for comments

RFC20050927ARa: Clarification of infection timing

1st draft: A. Reeves and N. Harvey, September 27, 2005

Applies to: Model description v1.0.2a (June 11, 2005)

Type of change: Language clarification

Summary: The proposed change explicitly codifies the current behavior of the model regarding the timing of infection events.

Change: This change applies to Section 3 (Disease), second paragraph. Proposed new text is highlighted:

> A unit can spend zero time in a state. For example, the parameter for time spent as Infectious Subclinical can be zero. In that case, units will change directly from Latent to Infectious Clinical. A unit undergoes its first transition state change on the day immediately following its infection.

End of changes

Justification:

This behavior follows naturally from computational requirements associated with state-transition models. It also addresses a systemic problem identified in legacy versions of SpreadModel.

State-transition models could potentially be coded in one of two ways: changes could be applied individually to entities in the model as they occur, or they could be stored and applied to all entities at the end of each time step. The following example illustrates the flaw in the first of these two approaches:

Consider 3 herds, with one infected herd.

1 2 3
L S S

The computer program, which takes actions step-by-step, processes the herds in order:

Processing herd 1...
Herd 1 infects herd 2. Herd 2 is immediately changed to infectious.
Processing herd 2...
Herd 2 infects herd 3. Herd 3 is immediately changed to infectious.
Processing herd 3...
Nothing more to do: the simulation day ends.

So by the end of the day, infection has spread to all herds.

Now consider the same situation, but with herds listed in a different order.

1 2 3
S S L

Identical processing would result in the following sequence of steps:

Processing herd 1...
Nothing to do.
Processing herd 2...
Nothing to do.
Processing herd 3...
Herd 3 infects herd 2. Herd 2 is immediately changed to infectious.
Nothing more to do: the simulation day ends.

At the end of the day, herd 2 is infected but herd 1 is not. The arbitrary re-numbering of the herds has created a different outcome.

This kind of situation arises whenever a serial program is used to simulate changes that actually occur in parallel. To avoid situations analogous to this one, the following sequence of steps is taken:

1) A decision is made regarding the state of each entity at time step *n*.
2) Decisions are made regarding the changes/effects that will occur.
3) Finally, the states of all entities are updated *en masse* for time step *n+1*.