ISCB/ISMB 2006 Fortaleza, Ceara, Brazil Academic Software Demo

"Cancer knowledge exploration and modeling with the *Oncology Thinking Cap*"

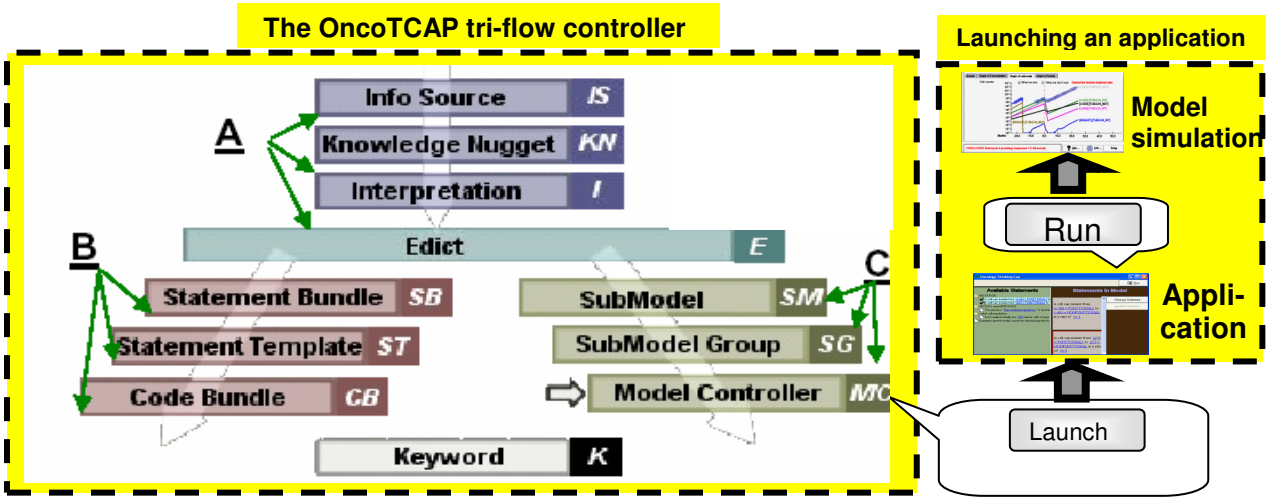
Roger Day, Bill Shirey, Michele Morris, University of Pittsburgh

OncoTCap tightly integrates a cancer knowledge base with an agent-based simulation engine. It synthesizes highly heterogeneous information to create and calculate a model. OncoTCap includes a complete methodology for building and validating natural history models of cancer from the molecular network through levels of scales up to the clinical trial. A major criterion is to facilitate the work of people with a diverse set of skills and goals, so that teams can work together effectively. The program's architecture is reflected in the main navigation tool, the "Tri-flow" browser-controller. The browser enables one to view a complex knowledge acquisition and modeling effort from many different perspectives. As seen in the figure below, the lynchpin (middle) is the "**Edict**", holding free-text instructions for running a simulation. Edicts can direct the behavior of biological agents, stipulate an experimental design or setting, or require execution of a goodness-of-fit test for validation. Edicts are shared by three work flows:

A The ***knowledge acquisition*** *work flow* supports recording **InformationSource**s, acquiring and assessing **KnowledgeNugget**s, attaching free-text **Interpretation**s, and constructing free-text **Edicts** for building models.

B The ***code-mapping*** *work flow* involves searching a catalog of "**StatementTemplate**s", each of which represents a simple or composite idea as a sentence with "blanks" or parameters, and then representing the **Edict** by selecting a pre-existing **Statement-Template** and "filling in the blanks". Each **Statement-Template** has previously been tied to Java code.

C The ***application building*** *work flow* involves successively grouping selected **Edict**s to create "**SubModel**s", "**SubModelGroup**s", and finally "**ModelController**s". **ModelController**s contain specifications for auto-generating modeling applications, such as validation suite testbeds, treatment optimization routines, and patient simulators for professional training. From a selected **ModelController**, a ***Launch*** button automatically generates and runs a new application program. In the application program, once the user makes final model selections, a ***Run*** button causes a model to be computed. That is, a pre-compiler writes Java source files, which are compiled, and finally the simulation or calculation is begun.



In this demonstration, we will step through the three work-flows and execute the resulting model application for the example of a model of mitosis, mutation, and apoptosis. The behaviors of a cell are represented as functions of internal states describing the current degree of functionality of each molecular component, each of which depends on the genetic complement, expression levels, and evolving micro-environment. Time permitting, we will also demonstrate how the architecture supports these features:

- How natural language is used to represent and re-use biological and clinical ideas.
- How to incorporate an expanding set of knowledge acquisition helpers.
- How to use other modeling engines besides the native simulator.