> In practice, do you foresee usage like short (reduced) checkings during work day/development and long (more exhaustive) checkings overnight or even over a full week?

Yes, we have a concept of a "scenario" which defines, among other things, a subset of the model and the applicable Stop Conditions, Observers and Verification MSCs. Users may define "full", "exhaustive" scenarios and "smoke" (using test nomenclature) scenarios which e.g. split the entire model into smaller partitions, cull certain exploration paths using Stop Conditions, and apply just some of the requirements. Such split is a user decision. We would like to make it possible to integrate the tool with Continuous Integration and build systems, so that e.g. certain checks could be run during development, possibly on the developer's machine, and certain checks could be performed by CI before merges, or milestones.

> In slide 16 you mentioned complexity reduction. Do you have a filter / refinement of the input, in order to reduce the generated state space in promela? Or does cover the entire SDL semantics?

We foresee refinement of the input via ASN.1 type specialization, state collapse via Observers and exploration path culling via Stop Conditions. As for SDL, we plan to transform most of the SDL semantics supported by OpenGEODE (there are some Promela limitations), reporting errors if the transformation is not representative. We do not plan any simplifications. Please note that in TASTE the "state space" of an SDL machine is also defined by the applicable ASN.1 definitions of the used types.

>Is SDL suitable to represent the application model with physical components?

Not in the general case, but the primary focus of the model checker is the logical correctness of the control application. In that respect, physical components can be usefully represented by approximations. The behaviour of those approximations need only be as rich as is necessary to completely explore the path space of the control application. In the case of refueling, the feared events from the controlsystem design point of view are configurational rather than dynamic, e.g. bad combinations of valve commands, so the approximation of the physical component behaviour can be fairly simplistic.

>Do you adopt the SDL semantics for composition of components (based on ASMs)? How will you validate the translation into promela?

In TASTE, each component ("TASTE Function") may have a single SDL process. Such process is a single state machine, with possibly nested and parallel states. We plan to support most of the SDL semantics supported by OpenGEODE (there are some Promela limitations), reporting errors if the transformation is not representative. TASTE Functions are connected as defined in the Interface View, from which the actual Promela processes will be derived, thus supporting multiple concurrent components. The correctness of the translation functions will be verified by unit and integration testing.