- David E Wilkins home page: http://www.ai.sri.com/people/wilkins
- His publications: http://www.ai.sri.com/~wilkins/bib.html
- SIPE home page: http://www.ai.sri.com/~sipe/

I am David Wilkins, a Senior Computer Scientist at SRI International, a not-for-profit contract research institute in Silicon Valley. I finishing my PhD at Stanford in 1979, my adviser was John McCarthy, who taught my first LISP class. I immediately went across El Camino to work in SRI's Artificial Intelligence Center, primarily because Nils Nilsson was there.

My first boss at SRI was Earl Sacerdoti, the author of NOAH. Before NOAH, AI planning had been a search in the state space. An example of this is STRIPS, developed at SRI for Shakey the Robot. NOAH introduced plan-space search into AI planning.

Inspired by Earl and NOAH, I became interested in solving real-world planning problems and found that this required the use of more domain knowledge and more expressive representations.

I divide past research in AI planning into two camps:
(1) systems that take a minimalist approach to domain knowledge (using STRIPS-style descriptions of primitive actions). "primitive-action planning"

(2) systems that focus on leveraging as much domain knowledge as possible. I focused on this approach in developing SIPE (System for Interactive Planning and Execution).

SLIDE: Oil Boom around island
Example domain that was encoded in SIPE: planning responses to oil spills.

The picture shows boom being placed around an island to protect if from an oil spill.

Note that there are multiple agents executing the plan, and that getting sufficient boom to surround the island is important.

SLIDE: Techniques developed in SIPE for Real-world Problems
These are Examples of increased expressiveness and techniques for making reasoning more efficient. This list was driven by client needs to solve real problems. Some of these techniques are now called "hierarchical task-network planning".

Multiple abstraction levels -- Many real problems have distinct natural abstraction levels, making them natural fits for HTN planning. Also, it is a powerful way to control search, and for humans to interact with the system. It can be hard to understand a primitive-action plan at the lowest level of detail

Parallel actions - Realistic domains can have dozens of (perhaps necessarily) parallel activities, as activities of various agents are coordinated. Parallelism can cause computational problems, and some systems produce only sequential plans.

Context-dependent effects -- Realistic domains often have numerous context-dependent effects. For example, when you move an object, everything attached to it or on top of it also moves to the new location. SIPE would deduce these context-dependent effects, but in STRIPS one might need an operator for every possible combination of objects attached to or on top of the moved object. This can cause an exponential explosion in the number of STRIPS operators needed.

Constraints ; Resources
Reasoning with numbers is essential in every realistic domain that we have studied. Common needs for numbers are time, resources having a specific capacity or available in limited quantities, and goals of accumulation. An example of goals of accumulation is obtaining a certain quantity of resource that must be assembled from smaller aggregations, such as getting enough boom from several warehouses to contain an oil spill.

Heuristics and representations to efficiently reason about actions

Replanning during execution – in the real world, execution rarely proceeds as planned, so SIPE must be able to replan during execution. Many AI planners do not provide such a capability.

Interactive, menu-driven, graphical interface – SIPE had perhaps the most advanced GUI in AI planning in the 80s and 90s. Interacting with people is a critical aspect of real-world planning. Realistic problems are embedded in the world and generally do not have precisely defined boundaries or evaluation functions. It is also hard to specify when a situation warrants breaking rules or ignoring certain information, yet such situations are common in real life. In such cases, a human user must be able to guide the planner and evaluate the plans produced, allowing the planning system to take advantage of the user's expertise.

The GUI also allows the user to understand large plans by viewing them at various abstraction levels, drilling down for details as needed.

SLIDE: screen shot of oil-spill response plan.
This slide shows a screen shot from the SIPE interface of an oil-spill response plan. Looking at the top 3 nodes, the blue oval is an action to deploy 3000 feet of boom to the Berkeley Eel Grass at time 3. The aqua colored hexagon before it is a goal that still needs to be solved at the next hierarchical level of expansion, to get the boom to Berkeley.

Finally, note the goal just below these 2. In fact, the higher-level goal was to get 9,000 feet of boom there, and SIPE split this into 2 actions, getting 3,000 from one location (a fact known to SIPE) and then posting in parallel a goal to get another 6,000 feet. The entire plan had a few hundred actions when completed.

Slide: HTN in SIPE
The final slide shows SIPE in operation. It takes 3 inputs, on the left, representing the world (such as the knowledge of where the 3000 feet of boom was located), the operators, which represent the actions that can be taken at multiple levels of abstraction, and the goals, which are to be achieved by the plan. Then the Plan Generator search though this space, applying operators to achieve goals. As depicted on the right side, the expansion goes level by level in the hierarchy until an executable plan composed of only primitive actions is obtained.

The plan is given to the Executor and if something unexpected happens, the Executor calls the planner again with a new goal and new information about the world state, so that SIPE can repair the plan.

This concludes my brief overview of the motivations behind SIPE.
For more information, see the SIPE home page, and a page with my publications.

Thank you for your attention.