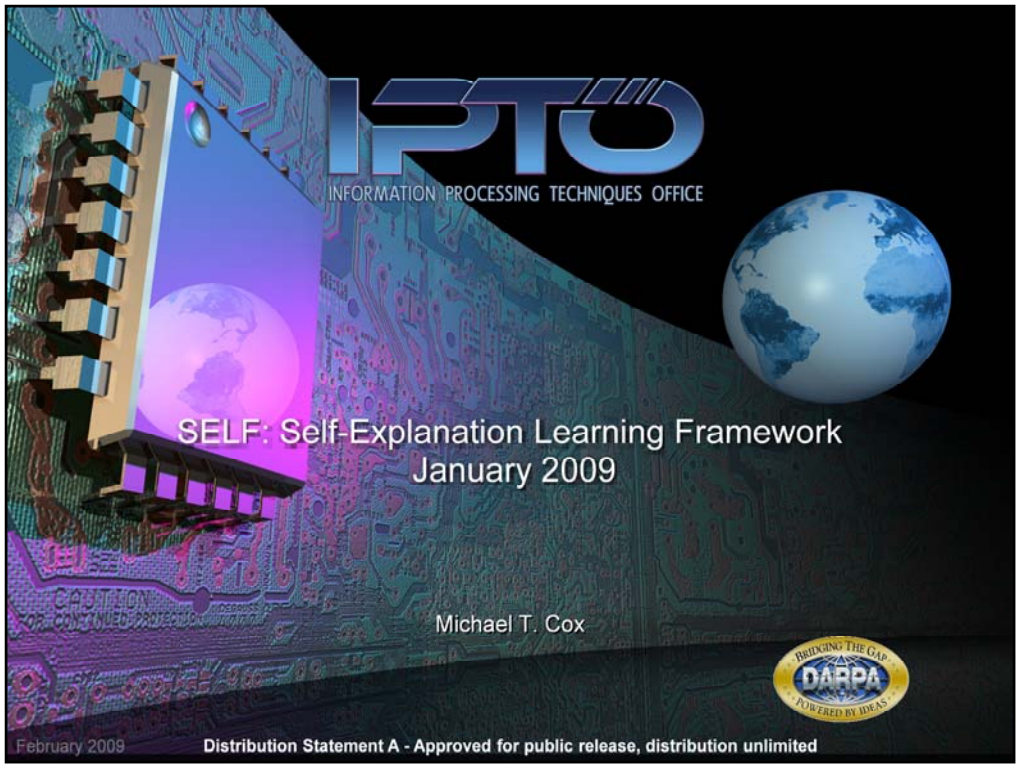
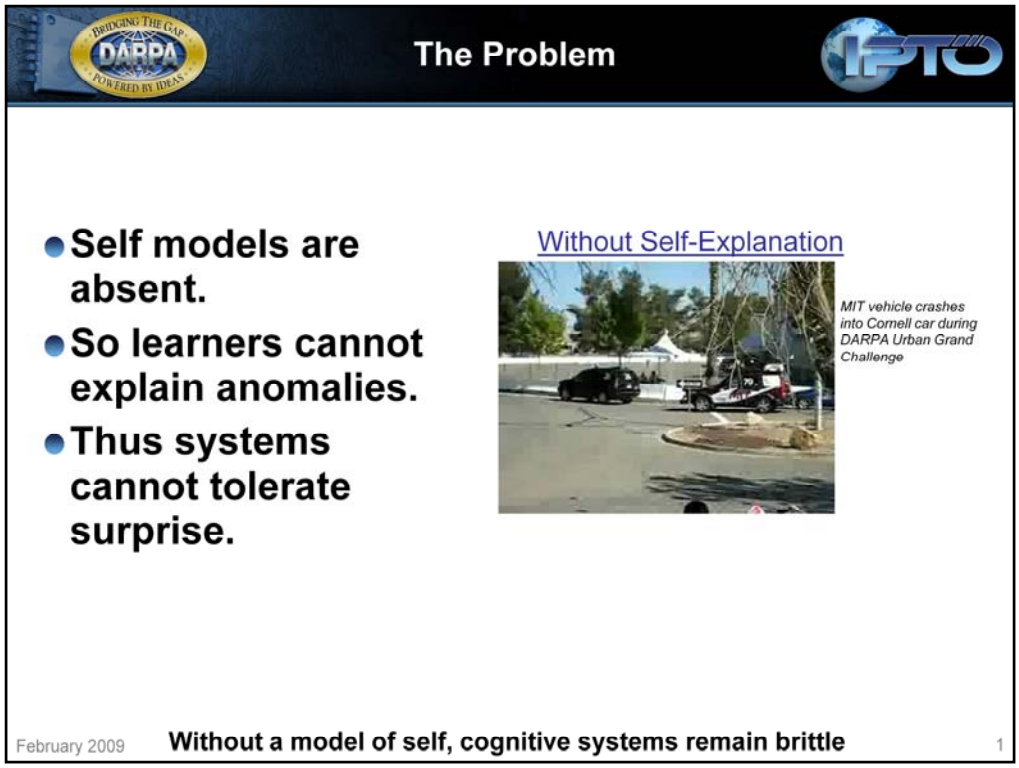



**The major claim**: Without metareasoning and metaknowledge, a cognitive system is not intelligent because it lacks explicit self-knowledge.

Elaboration: It does not really know what it is doing nor what its own capabilities are. Its learning is limited, its interactions with users are unnatural, and it cannot explain itself. Because it cannot explain itself and its failures, it will remain forever brittle in the face of surprise. To do any of this intelligently, it needs to learn an explicit model of itself and reason about the model. Metareasoning goes to the heart of what it means to be intelligent.

Many DARPA programs, especially in IPTO, have a meta level in its architecture or some portion of its effort aimed at metareasoning. However all programs end up focusing on building yet another cognitive agent. No other program provides as government furnished “equipment” a mature performance agent for all contractors so that the focus remains entirely at the meta-level.




Systems cannot learn from surprise, because they have no self-model from which to explain anomalies and hence generate solutions. Self-explanation fills this gap.

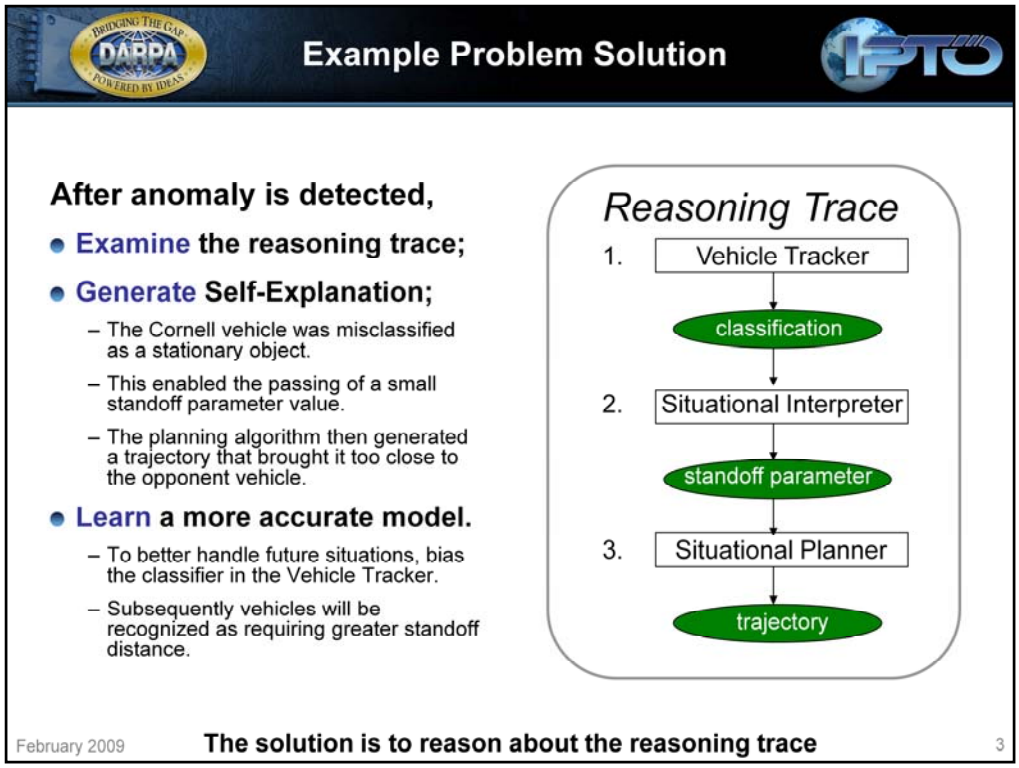
During the DARPA Grand Challenge the MIT vehicle collided with the Cornell vehicle. Afterwards the MIT developers and analysts examined the failure and explained what caused the error by identifying the causal factors responsible. They then used this explanation to determine what needed to be changed in the control program. Programmers then implemented the change. THIS IS WHAT WE WANT THE SYSTEM ITSELF TO DO.

*What are you trying to do? Articulate your objectives using absolutely no jargon.*

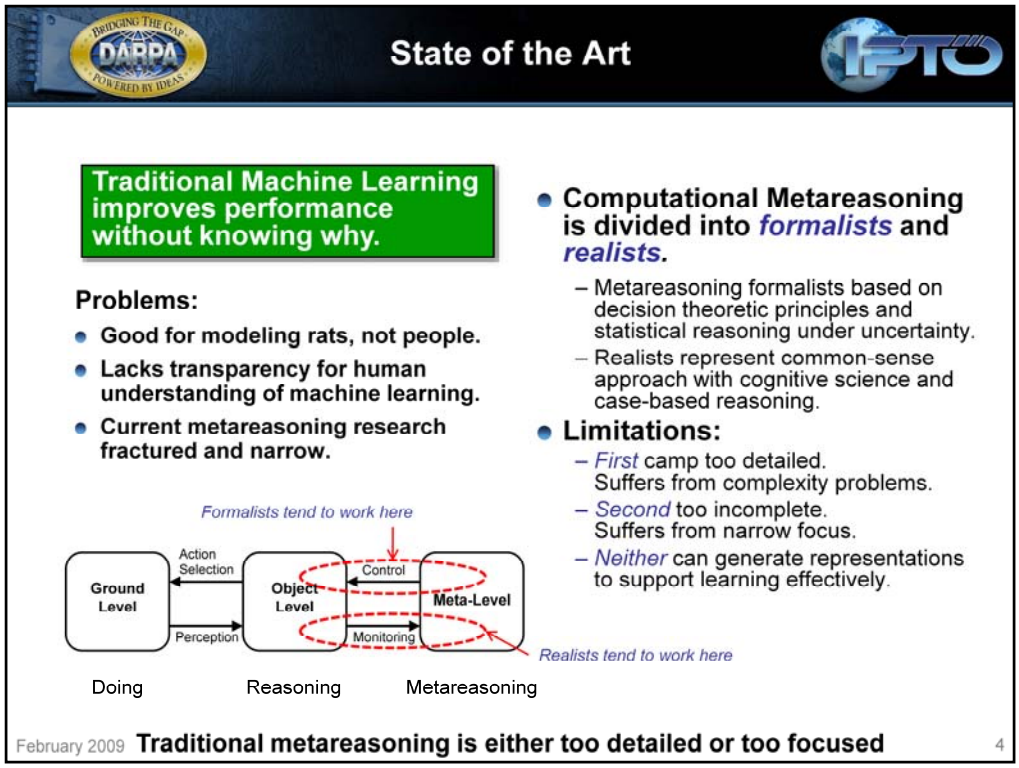
The SELF Program will enable any learning system* to explain and repair itself.

*That is, any learning system that conforms to the metacognitive API and exposes its knowledge, input, goals, and strategies.

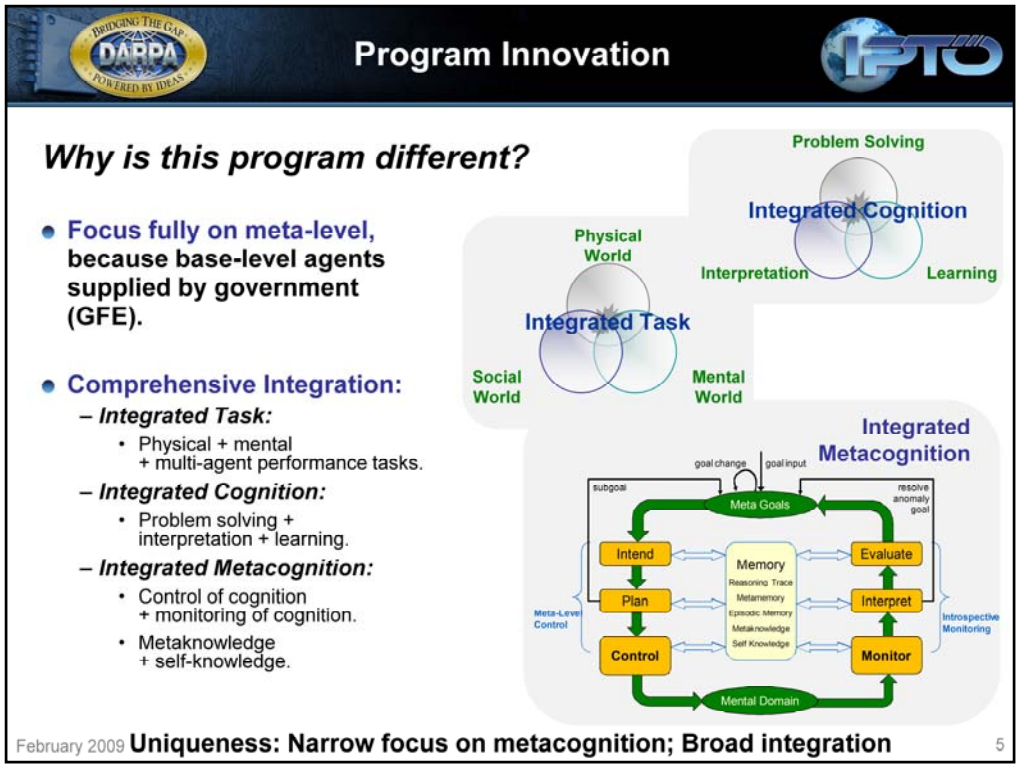
Although many benefits flow from this capability, it will result in two specific program deliverables. First we will construct a general purpose module that can improve any cognitive system that conforms to the API. Second we will develop a general system that helps engineers build all kinds of cognitive systems.




The immediate cause of the collision (slide 1) was the trajectory that brought the MIT vehicle too close to the Cornell car. However by examining the trace of the reasoning preceding the incident, the observer can conclude that the actual cause was the Vehicle Tracker that classified the car as parked and stationary. Thus the learning can target the right cognitive module for learning a more robust solution.




*How is it done today, and what are the limits of current practice?*

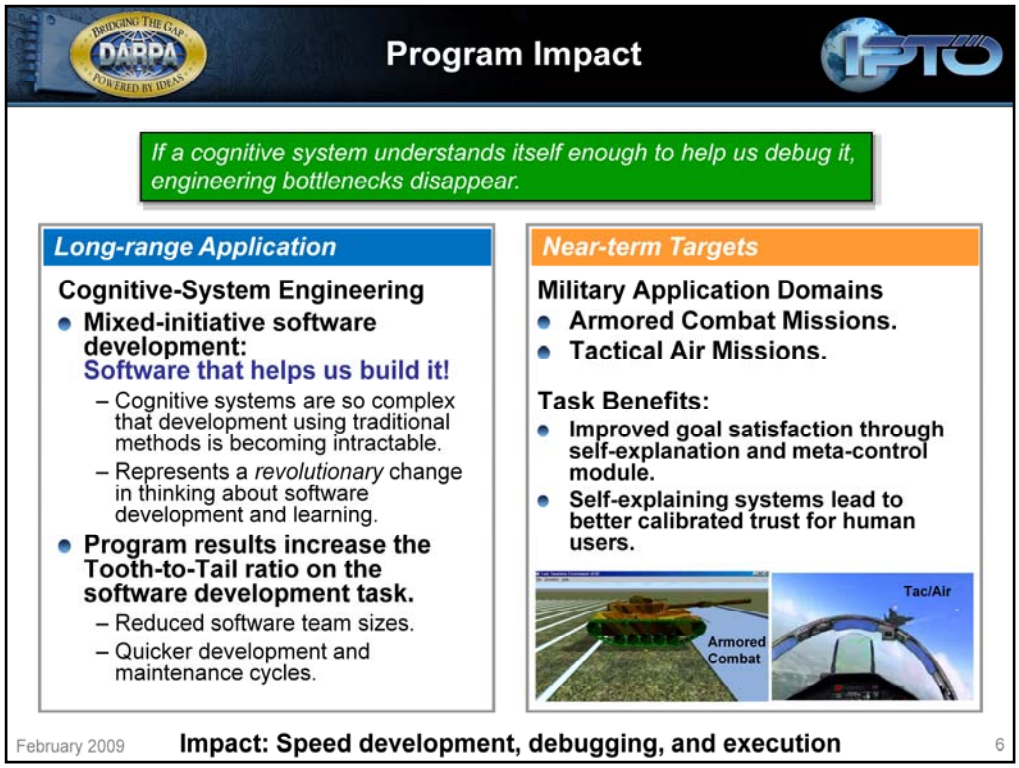



*What's new in your approach and why do you think it will be successful?*

Narrow focus: Many IPTO programs spend the bulk of the time and funds building cognitive agents. We intend to supply the agent as GFE and focus on metacognition rather than cognition.

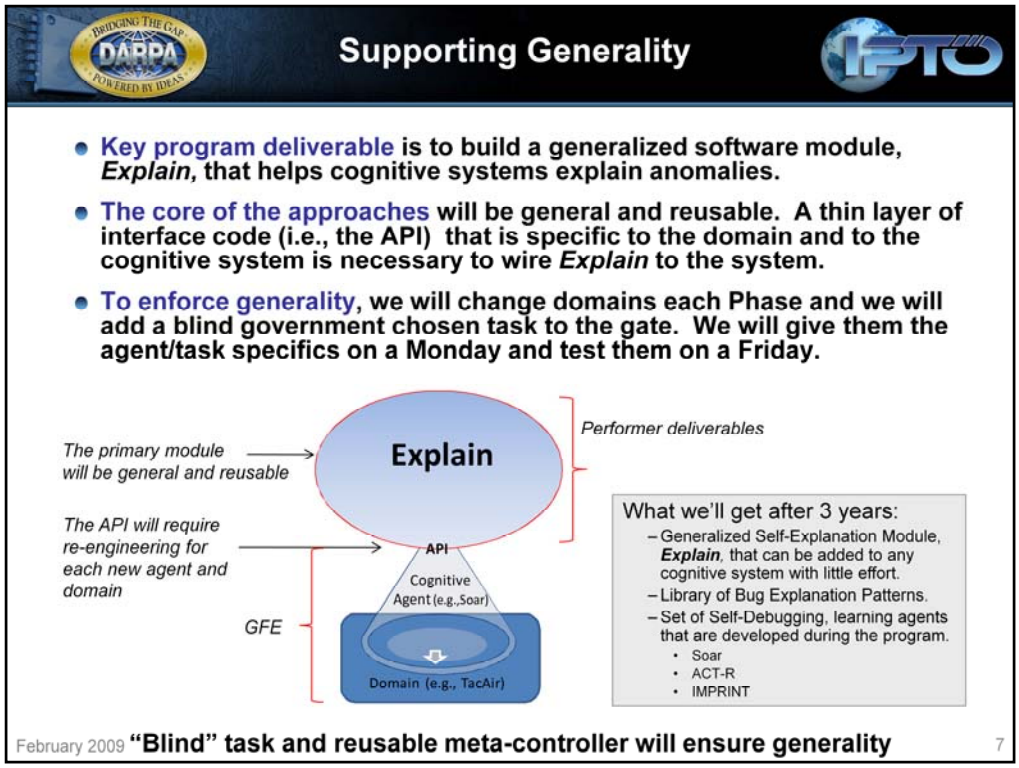
Broad comprehensive integration: No one has ever attempted to pull it all together.

This research is ripe for exploitation, because there currently exists work on parts of the problem by many researchers and a convergence of theory and applications has begun. We can build upon the exceptional work on critical representations and algorithms that has begun at various research institutions.



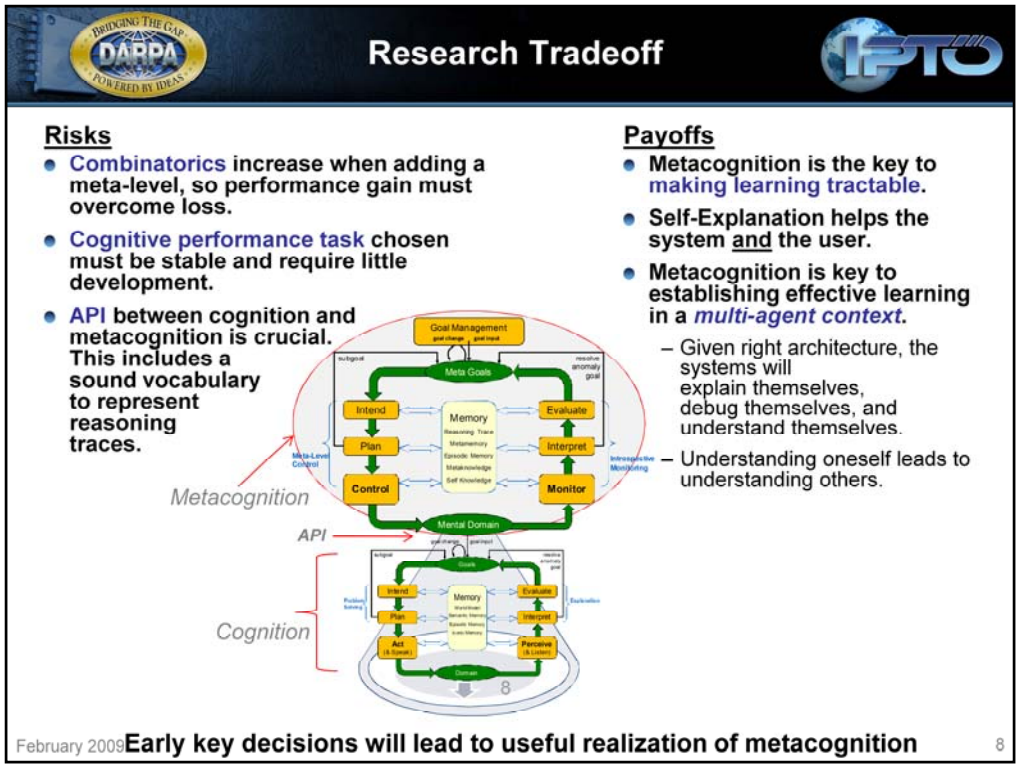

*Who cares? If you're successful, what difference will it make?*

Ultimately if this program is successful, we can make engineering cognitive systems quicker and more productive, using fewer engineers. The results will enable efficient development and debugging of intelligent agents, because the technology will enable a system to participate in its own development cycles. In the shorter term, this technology will allow metareasoning components to be bolted onto any intelligent system that conforms to minimum guidelines, thus improving the performance of that system. The program impact is therefore twofold: to improve the engineering of cognitive systems and to increase their performance.

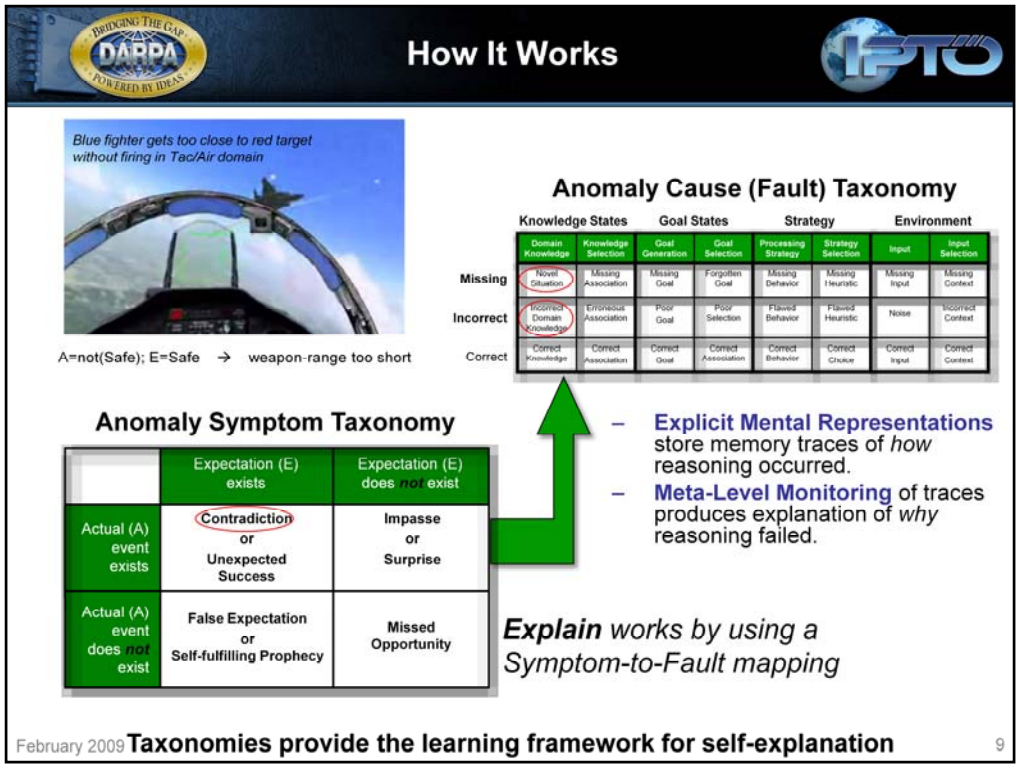



*How can the SELF Program results apply outside of the domains under which they are demonstrated. That is will the systems have to be totally rewritten for new applications in the future?*

We will not construct domain-specific solutions. Rather we will create a general module called Explain that can be bolted upon any agent in any domain as long as the agent implementation conforms to the API's constraints. Explain remains constant; the API will be modified for each new agent architecture and each new domain.




*What are the risks and the payoffs?*



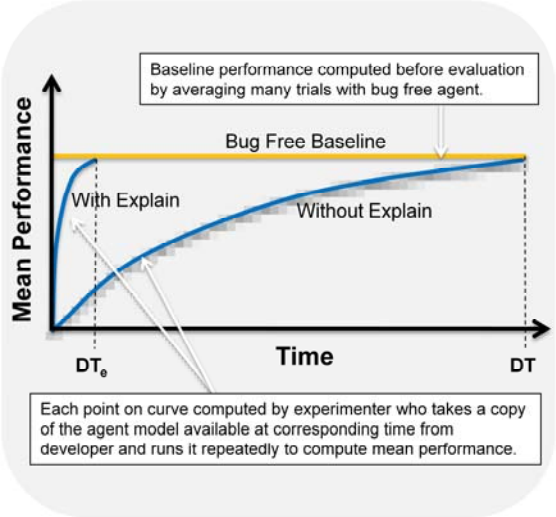

Diagnosis can be cast functionally as a symptom-to-fault mapping. So can self-diagnosis (i.e., self-explanation of anomalies) if one has a framework with which to represent symptoms and faults in the cognitive domain. The two taxonomies here provide the *self-explanation learning framework*.

In the Tactical Air domain the agent pilots a fighter too close to red without firing. The agent had thought it was safe when it was not, and thus the symptom of the anomaly is a contradiction. The self-explanation task is now to map from this symptom to the underlying causes of the anomaly by examining a trace of the prior reasoning that preceded the incident. The weapon range may have been set at 5km instead of 50km, so the cause is both incorrect knowledge (5km) and missing knowledge (50km).

## Research Assessment

- Metric is ***Reduction in Debugging Time***.
- Technology developed under SELF will lead to a ***tenfold reduction*** in cognitive-agent debug time, while restoring baseline domain performance.
- This improvement will be shown for agents having multiple, interacting bugs across *all* bug categories.

- **DT** = Manual Debugging Time.
- **$DT_e$** = Debugging Time with Explain.
- **Reduction of Debugging Time = $(DT-DT_e)/DT) >= 90\%$.**




February 2009 **Research promises an order of magnitude improvement.** 10

# Phase Gates

## Go/No Go Overview

- **DARPA provides cognitive agent that can achieve goals in domain.**
- **Central task is to debug agent.**
  - **Manually**: Human debugging and code repair without *Explain* component.
  - **Mixed-initiative**: *Explain* aids human in testing, analyzing, and fixing agent system.
    - **Self-explanation** maps performance problems to bugs at knowledge level.
    - **Human uses explanation** to better understand code level and make fix.

## Experiments

- **For each trial (problem),** experimenter inserts into agent model **multiple, interacting bugs from *any* bug category**.
- Humans tested in two conditions: with and without *Explain* turned on.
- **Reduction of Debug Time averaged across multiple trials.**
- **Contractors compete for greatest Reduction of Debug Time above minimal threshold.**

| | Phase I | Phase II | Phase III |
|---|---|---|---|
| Number of Domains | 1 | 2 | 3 |
| Bug Types | 4 | 4 | 4 |
| Objective Function returns to baseline | 100% | 100% | 100% |
| Reduction in Debug Time | >= 30% | >= 60% | >= 90% |

NEW DOMAIN EACH PHASE

Armored Combat

Tactical Fighter Missions

Time Critical Targeting (AOC)

BUG TYPES

Knowledge

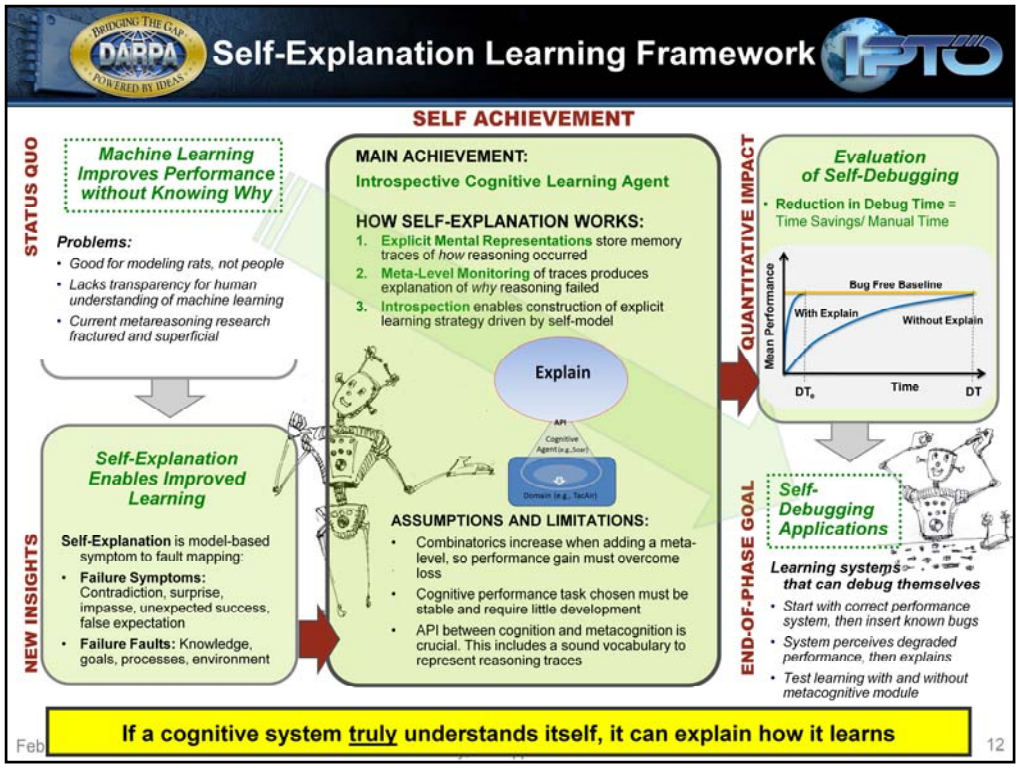
Goal

Strategy

Environmental Input



**DARPA-hard problem; Simple metric.**




Penta chart for SELF

## QUALIFICATION

**The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.**

Approved for public release, distribution unlimited.
**Contact Michael Cox (Michael.Cox@DARPA.mil).**