Agent modules are formed based on Core Competencies of tasks.

After implementation, a run-time trace of the agent is recorded.

Graphs are generated from the run-time trace and are used to verify the architectural model.

Each function is decomposed into tasks, which are associated with core competencies.

Developer selects functionalities for inclusion in the agent.

Desired Agent Functionalities

Dynamic Organization Selection;
Declarative, Behavioral, & Intentional Modeling;
Collaborative Planning; Etc.

CC-based Task List

Dependency Graph

A tool suite for agent technology developers to design and analyze agent systems through the development lifecycle from architectures to implementations.

A tool suite for agent technology developers to design and analyze agent systems through the development lifecycle from architectures to implementations.

Copyright © 2002 by The University of Texas at Austin, The Laboratory for Intelligent Processes and Systems
Designer’s Agent Creation and Analysis Toolkit (DACAT)

K. Suzanne Barber and Dung N. Lam
The Laboratory for Intelligent Processes and Systems
The University of Texas at Austin, Austin, TX 78712-1084
{barber, dnlam}@mail.utexas.edu

INTRODUCTION
The problems addressed in this research are the lack of interoperability among agent systems, the lack of common design and evaluation methodologies, and the lack of a common definition for an agent or multi-agent systems. The goal of this research is to formalize the capabilities that should be expected of agents as well as the interdependencies between these capabilities in order to establish a foundation on which all agents can be designed and built. To encourage the rapid development of interoperable, compatible, and flexible agents, tools based on disciplined design and evaluation methodologies will be produced to assist designers in constructing agent architectures and evaluating design decisions.

CORE COMPETENCIES
An agent is characterized according to its functionalities, which can be grouped under six core competencies: sensing, modeling, organizing, planning, acting, and interoperating. In fact, all agents, including reactive agents, must implement each of these core competencies explicitly or implicitly. The science for selecting the required core competencies, understanding the inherent dependencies between those core competencies as well as understanding and evaluating techniques for implementation remains a research issue.

DACAT’S DESIGN SUITE
With the foundational concept of core competencies established, a toolkit called DACAT is being developed to facilitate the design and evaluation of various agent designs. The first tool, called TLReader (Task List Reader), is used to help the designer build an agent architecture based on core competencies from a list of tasks, which were decomposed from the desired functionalities. Each task has several attributes, including input, output, interaction, and the core competency to which the task is related. The list of tasks is translated into XML, which the TLReader deciphers and visualizes as nodes and connections. Nodes represent tasks, resources, and interactions, and the directed connections represent the nodes’ dependencies with other tasks, resources, and interactions.

The TLReader tool also helps the designer organize the agent architecture on the basis of core competencies. Tasks that are related to the same core competency cluster together, while tasks that are not related drift graphically apart. As a result, these clusters of functionality can be grouped together as a core competency module within the agent. By using the core competency modules as the agent architecture, the dependencies among module should be minimized by appropriate designer selections. As a result, the plug-and-play of strategies is simplified because all dependencies are more easily identified. Recognizing these dependency among tasks, resources, and interaction gives the designer a better understanding of the inner workings of the agent.

The core competencies-based organization is an intuitive abstraction of the functionality of an agent. Existing agents can be evaluated using core competencies by defining all the tasks the agent is capable of and using the TLReader to visualize the core competency-based architecture. Thus, evaluation of any agent is facilitated and comparisons among agents with different architectures are possible. At this stage, evaluation consists of an analysis of the dependencies within an agent and of the flexibility of the agent’s architecture.

DACAT’S VERIFICATION SUITE
In the developmental stages of the agent, an analysis of the agents codebase can be used to maintain consistency between model and implementation. DACAT’s Tracer tool is used to debug, analyze, and enhance the agent system using code profiling and timing data. From the Tracer’s logged runtime data, a graph can be inferred and automatically generated, enabling design and architectural improvements to be done based on the actual implementation rather than just the initial model. The results of the tools in the verification suite provide detailed automatic documentation using the logged data and inferred models.

CONCLUSION
The benefits of establishing the concept of core competencies include 1) sharing a common characterization of agents and having a measurement of agency, 2) a basis from which to derive methodologies for agent design and evaluation, and 3) the ability to modify a component of an agent with complete knowledge of its affects on other components of the agent. By providing tools for the creation and modification of agents, rapid development and interoperability among agents is facilitated.