CSC/CPE 481 Final Exam

Winter 2003

Prof.: Franz J. Kurfess

This is the final exam for the CSC/CPE 481 Final Exam, Winter 2003. It is a take-home exam, and you may use textbooks, course notes, or other material, but you must formulate the text for your answers yourself. You are not allowed to discuss the questions and answers with other students or anybody else. If you need clarifications about questions, you can contact me via email, or see me during my office hours on Monday from 3-5 pm and Tuesday from 3-6 pm. The deadline for the exam is Tuesday, March 18, 2002, at 6 pm. You must submit a printed and signed copy of the exam, which you can either leave in the drop box in front of the CSC department office (room 14-154), or give to me on Tuesday, March 18 during my office hours.

Student Name:	**Date:**
Signature:	

Task I – Short Questions

1. Discuss the suitability of CLIPS for the two following example problems. Describe the strengths and weaknesses for each example, and determine if you would use CLIPS for such a problem, or not. Identify the crucial aspects for your decision.

   (a) Intrusion Detection: A program that observes relevant activities on a computer system or within a network, and raises an alert when it detects suspicious activities. 10

(b) Document Categorization. A program that inspects the contents of text-based documents, and assigns one or more categories to a document. The categories can be defined by an ontology, for example. You can assume that the documents are from a specific domain, e.g. "Artificial Intelligence," and that filters are available that convert other formats into plain ASCII text documents. 10

2. What are the main differences between *sound* and *unsound* inference methods? Give at least one example for each category. 10

   (a) Sound inference methods:

   -
   -
   -
   -

   Example:

   (b) Unsound inference methods:

   -
   -
   -
   -

   Example:

Task II – Software Component Selection System (SCSS)

In this task, you need to describe the design of a knowledge-based system that provides assistance for computer-supported selection of software components based on requirements specifications. You can assume that your company has access to a very large library of thousands of software components. These components are accompanied by documents that describe their purpose, structure, API, relationship with other components, and other relevant aspects (e.g. implementation language, performance data,) in a standardized format. This format has the following main elements:

Purpose: Description of the intended use

Interfaces: Specification of interfaces between components

Naming: Unique names for components and interfaces

Behavior: Description of the behavior of the component

Meta-data: Relationships between components and interfaces

Interoperability: Communication and data exchange among components from different vendors and with different models (e.g. CORBA, COM+, Java Beans)

Customization: Modification of interfaces, behavior according to customer requirements

Composition: Interfaces and rules for combining components into larger structures

Evolution Support: Rules and services for replacing components and interfaces by newer versions

Packaging and Deployment: Required resources and services for the installation and configuration

"Description" in the above list indicates that natural language is used, whereas "specification" indicates the use of a formal method.

The problem is that it is very difficult for a developer to identify the most suitable component for a particular purpose. Due to your experience with expert systems, you have been assigned to do a feasibility study for a system that takes a high-level specification written by a developer, and identifies components from the library that might be suitable for use in an implementation.

The system must provide at least the following functions:

Name-Based Retrieval: Finds a component by its name (in case the developer knows it).

Interface Compatibility: Verify that the components selected are compatible with the interfaces as described in the specification.

Purpose Matching: The system should identify components that have a similar purpose to the one stated in the specification. One problem here is that the statements of purpose are not formally specified, but stated in natural language.

Matching of Other Aspects: The developer can select other relevant aspects from the ones available in the component description, and specify certain criteria that are considered important. For the feasibility study, you are free to select one or more aspects, and you can make reasonable assumptions about the way the aspects are described.

Dependency Checker: Determines if a component requires other components.

You can assume that the component library is available as a large data bases.

a) Give a high-level description of your system, identifying the major components, and the way they interact. You can use a block diagram to illustrate this. 20

b) Briefly describe the knowledge base(s) and reasoning methods used by your system, and how they realize the functions listed above. 10

Name-Based Retrieval:

Interface Compatibility:

Purpose Matching:

Matching of Other Aspects:

Dependency Checker:

c) Given that funding would be a problem for the realization of such a system, do you think that expert systems are an appropriate technology for such a system? Would you recommend the use of CLIPS or JESS to implement the SCSS? Justify your recommendation! 10

40

Total Points: 70