

Requirements and Software Architecture for the CAT Assistant Tool:

Supporting the COTS-Aware Requirements Engineering and Software Architecting

(CARE/SA) Approach (version 1.0)

**Technical Report UTDCS-25-05
The University of Texas at Dallas**

**Michael McFadden
Kendra Cooper**

June 2005

Revision History

Version	Author	Comments
1.0	K. Cooper, M. McFadden	Initial release

Table of Contents

1. Introduction.....	1
2. Requirements Specification	2
2.1 CAT Diagrams	2
2.1.1 Agent Diagram	2
2.1.2 Goal Diagram.....	4
2.1.3 Use Case Diagram.....	6
2.1.4 Sequence Diagram.....	8
2.1.5 Class Diagram.....	9
2.1.6 Object Diagram.....	10
2.1.7 Collaboration Diagram.....	11
2.1.8 Impact Diagram.....	12
2.2 COTS Component Repository	13
3. Graphical User Interface	15
3.1 Main Form.....	15
4. Software Architecture	16
4.1 Logical View.....	16
4.2 Persistent Data.....	17
4.3 Concurrent Users.....	17
4.4 Distributed Processing	17
References.....	17

1. Introduction

The goal of developing systems better, faster, and cheaper continues to drive software engineering practitioners and researchers to investigate new software engineering methodologies. In requirements engineering, the focus has been on describing the functional requirements for systems that are being built from scratch. As the size and complexity of software systems continues to grow the use of commercial off the shelf (COTS) components is being viewed as a potential solution to this problem. Effective use of COTS components, however, requires a systematic methodology that would provide both a set of concepts for modeling the subject matter and a set of guidelines for using such concepts. In particular, the methodology needs to recognize and address the people oriented problems including the identification and resolution of conflicting goals as well as bridging the gaps between stated requirements and 'approximately fitting' components while still satisfying the customer.

The Component-Aware Technology (CAT) approach is focused on creating and modeling a requirements engineering approach that explicitly supports the use of commercial off the shelf (COTS) components (Cooper, Chung, Rampura, 2005). It is both goal- and agent-oriented, which helps to support the development of today's complex systems. The process is intended to assist the requirements engineer, not to replace their intelligence and experience.

The Component-aware technology (CAT) project is focused on defining a methodology that effectively supports the matching, ranking, and selection of COTS components in early phases of the software development lifecycle: requirements and architecture. The project aims to provide a set of models, techniques and tools including:

- A scheme for representing the various stakeholders as intentional agents (i.e., agents that make decisions), the high-level functional and non-functional requirements of such agents as goals, and the structural and behavioral system functional requirements as objects
- A scheme for matching, ranking and selecting COTS components with qualitative and quantitative decision making approaches. The CAT approach can be used to detect and resolve discrepancies between, the capabilities associated with the COTS components and the needs of the stakeholders associated with the current component based application
- Formal product- and process- models and meta-models that are consistent, complete, and correct
- Knowledge-based tool support that assists the developers in creating complete, correct, and consistent models by providing intelligent guidance and automated defect checking.

The purpose of this report is to document the requirements and software architecture outline for the Component-Aware Technology (CAT) Assistant Tool. The current versions of the CAT approach and tool support are targeted for the effective use of commercial-off-the-shelf (COTS) components, for which a blackbox specification of the components is available. In the future, the work will be generalized to support off-the-shelf components (OTS), for which a set of development models (i.e., goals, use cases, architecture, etc.) are available.

2. Requirements Specification

The CAT Tool is composed of two main sections. The first section is for modeling of component based applications. The second section is for modeling the COTS Component Repository.

2.1 CAT Diagrams

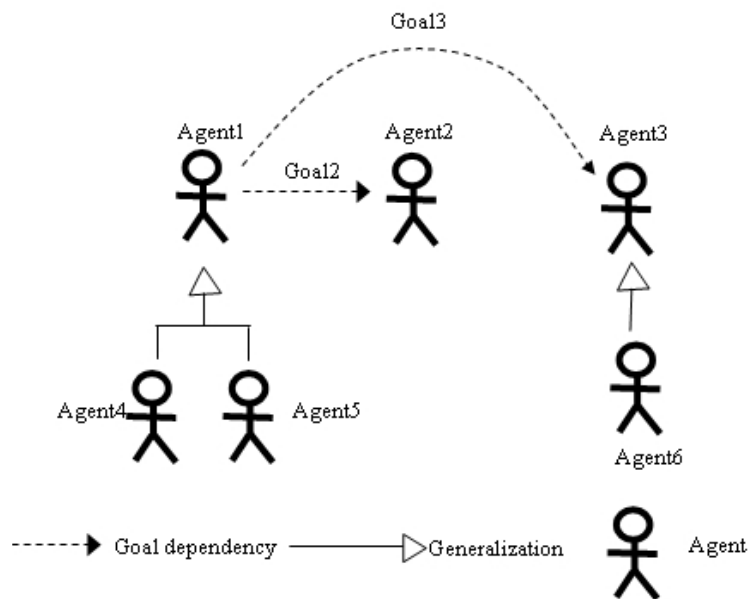
The CAT Tool is a CASE tool, which supports standard UML diagrams with extensions to support agent- and goal-oriented paradigms.

2.1.1 Agent Diagram

The Agent diagram allows the user to model the agents and their goal dependency associations for the system under development.

Visual Model

Agent Diagram Sample



Capabilities

- display Agent specific icons in the menu and on the toolbar (agent, generalization association, dependency association, note, anchor)
- display Agent View in main browser
- capability to display, create, modify, and remove Agent diagrams
- capability to display, create modify, and remove Agent objects
- capability to search for Agent objects
- add, modify, and remove generalization relationships (one Agent can be a generalization of one or more one or more Agents)
- add, modify, remove goal dependency relationships
- capability to add, modify, remove Notes

- capability to anchor a Note to an Agent
- store/retrieve Agent diagrams XML in ASCII file format
- store/retrieve Agent diagrams from RDBMS

Specification

Name	Cardinality	Type	
Type		string	1
Role		string	1
Association		string	1
Perspective		string	1
Relevance		string	1 or more
Generalization (parent agent)		generalization association	0 or more
Agent Goal (defined in goal view)		goal	1 or more
Agent to Agent Goal dependency		goal dependency association	0 or more
Abilities		string	1 or more
Assumption		string	0 or more
Notes		string	0 or more

- add, modify, remove goal dependency relationships
- capability to add, modify, remove Notes
- capability to anchor a Note to an Agent
- store/retrieve Goal diagrams XML in ASCII file format
- store/retrieve Goal diagrams from RDBMS

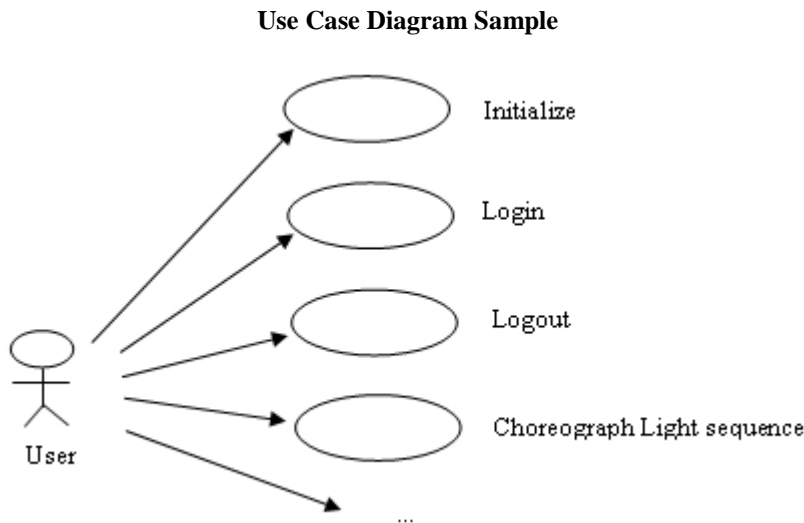
Specification

Name	Type	Cardinality
Type	string	1
Topic	string	0 or more
Priority	positive integer	1
Keyword	string	1 or more
Description	string	1
Level	positive integer	1
Traceability to use case realized by COTS	goal traceability association	0 or more
AND decomposition	COTS realization association	0 or more
OR decomposition	AND decomposition association	0 or more
Very positive contribution	OR decomposition association	0 or more
Positive contribution	very positive contribution association	0 or more
Neutral contribution	positive	0 or more
Negative contribution	neutral contribution association	0 or more
Very negative contribution	negative contribution association	0 or more
Assumption	very negative contribution association	0 or more
Notes	string	0 or more

2.1.3 Use Case Diagram

The Use Case diagram allows the user to model the actors, software requirements (use cases), and their associations for the system under development. The traceability relationships to COTS components that may realize the goals are also captured. The Use Case diagram is partially compliant with the Use Case diagram defined in Unified Modeling Language 1.5; it is not a complete implementation of the standard.

Visual Model



Capabilities

- display Use Case specific icons in the menu and on the toolbar
- display Use Case View in main browser
- capability to display, create, modify, and remove Use Case diagrams
- capability to search for Use Case objects
- capability to create, modify, and remove Actors
- capability to create, modify, and remove Actor Associations
- capability to create, modify, and remove Generalization Associations between Actors
- capability to add, modify, remove Notes
- capability to anchor a Notes
- store/retrieve Use Case diagrams XML in ASCII file format
- store/retrieve Use Case diagrams from RDBMS

Specification

Name	Type	Cardinality
Keyword	string	1 or more
Weight	integer	1 or more
Description	string	1

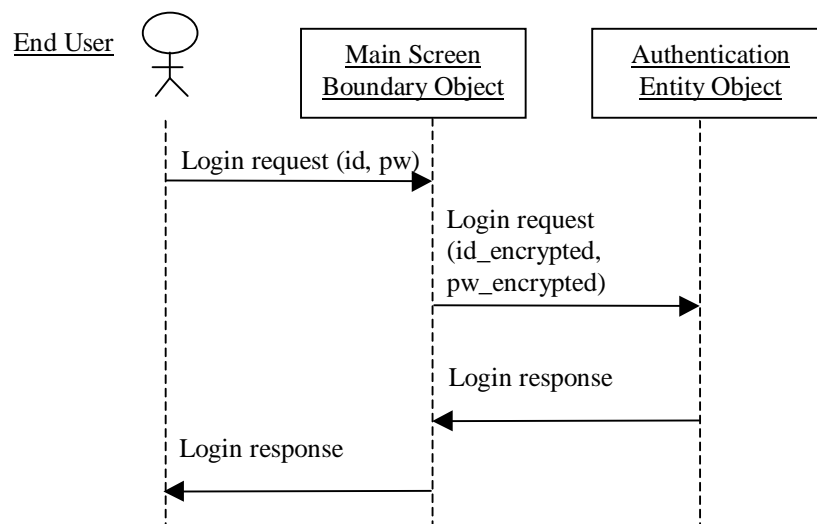
Traceability to Sequence Diagram	association	0 or more
realized by COTS	COTS realization association	0 or more
normal flow of event	string	1 or more
alternate flow of event	string	1 or more
included use case	use case name	1 or more
extends use case	use case name	1 or more

2.1.4 Sequence Diagram

The Sequence diagram allows the user to model the time ordering of objects and their associated method invocations for a section of the application under development. The Sequence diagram is partially compliant with the Sequence diagram defined in Unified Modeling Language 1.5; it is not a complete implementation of the standard.

Visual Model

Sequence Diagram Sample



Capabilities

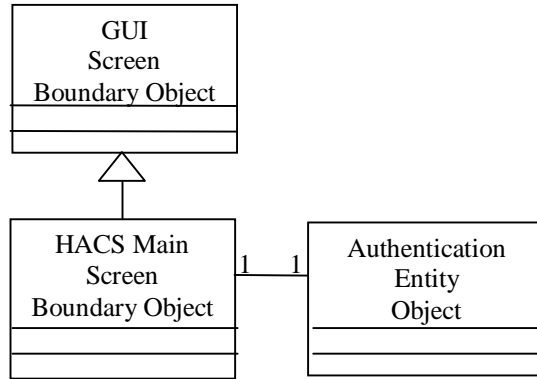
- display Sequence diagram specific icons in the menu and on the toolbar (objects, lifelines, different types of message passing, focus of control)
- display Interaction View in main browser
- capability to display, create, modify, and remove Sequence diagrams
- capability to display, create, modify, and remove Objects
- capability to display, create, modify, and remove Associations
- capability to display, create, modify, and remove Constraints
- capability to search for Objects and Messages
- capability to add, modify, remove Notes
- capability to anchor a Notes and Messages
- store/retrieve Sequence diagrams XML in ASCII file format
- store/retrieve Sequence diagrams from RDBMS

2.1.5 Class Diagram

The Class diagram allows the user to model the classes and their associations for the system under development. The traceability relationships to COTS components that may realize the goals are also captured. The Class diagram is partially compliant with the Class diagram defined in Unified Modeling Language 1.5; it is not a complete implementation of the standard.

Visual Model

Class Diagram Sample



Capabilities

- display Class diagram specific icons in the menu and on the toolbar (dependency, association, generalization, realization; with 2 variations of association: aggregation and composition)
- capability to display, create, modify, and remove Class diagrams
- capability to display, create, modify, and remove Relationships
- capability to search for Class diagram objects
- capability to group Class diagram objects
- capability to add, modify, remove Notes
- capability to anchor a Notes
- store/retrieve Class diagrams XML in ASCII file format
- store/retrieve Class diagrams from RDBMS

Specification (class)

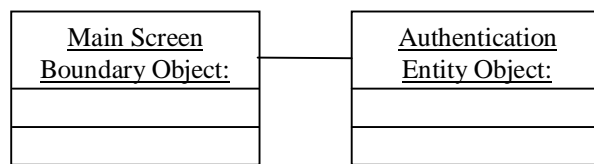
Name	Type	Cardinality
Class Name	String	1
Data attribute name, type, value, visibility	Data attribute	0 or more
Operations name, parameter values, return type	Operation	1 or more
Abstraction	string	1

2.1.6 Object Diagram

The Object diagram allows the user to model a set of objects and their relationships for the application under development. The Object diagram is partially compliant with the Object diagram defined in Unified Modeling Language 1.5; it is not a complete implementation of the standard.

Visual Model

Object Diagram Sample



Capabilities

- display Object diagram specific icons in the menu and on the toolbar
- capability to display, create, modify, and remove Object diagrams
- capability to display, create, modify, and remove Relationships
- capability to search for Object diagram objects
- capability to add, modify, remove Notes
- capability to anchor a Notes
- store/retrieve Object diagrams XML in ASCII file format
- store/retrieve Object diagrams from RDBMS

Specification (object)

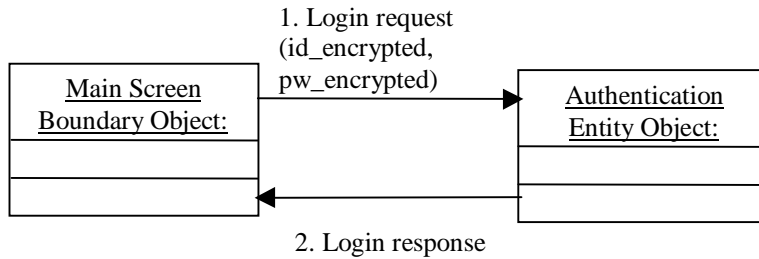
Name	Type	Cardinality
Name	String	1
Class Name	String	1
Data attribute	Data attribute	0 or more
name, type, value, visibility		
Operations	Operation	1 or more
name, parameter values, return type		

2.1.7 Collaboration Diagram

The Collaboration diagram allows the user to model the structural organization of objects and their associated methods for the application under development. The Collaboration diagram is partially compliant with the Collaboration diagram defined in Unified Modeling Language 1.5; it is not a complete implementation of the standard.

Visual Model

Collaboration Diagram Sample



Capabilities

- display Collaboration diagram specific icons in the menu and on the toolbar
- capability to display, create, modify, and remove Collaboration diagrams
- capability to display, create, modify, and remove Relationships
- capability to search for Collaboration diagram objects
- capability to add, modify, remove Notes
- capability to anchor a Notes
- store/retrieve Collaboration diagrams XML in ASCII file format
- store/retrieve Collaboration diagrams from RDBMS

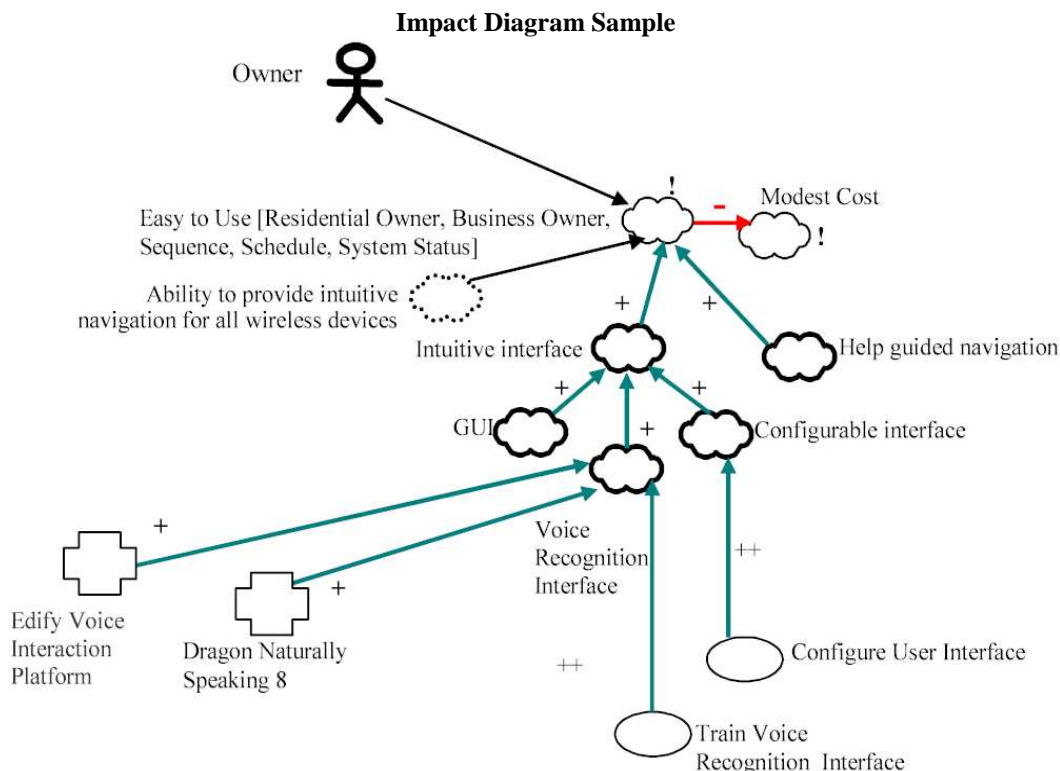
Specification (object)

Name	Type	Cardinality
Name	String	1
Class Name	String	1
Data attribute	Data attribute	0 or more
name, type, value, visibility		
Operations	Operation	1 or more
name, parameter values, return type		

2.1.8 Impact Diagram

The Impact diagram is a read-only browser that allows the user to graphically explore the traceability Relationships specified in the model. For example, the user can request to display all of the Agents in the model, select an agent, and then request to display all of the Goals related to that Agent. It is called an Impact diagram because it allows the user to determine the impact of making a change (i.e., an update or delete) to an element in the model.

Visual Model



Capabilities

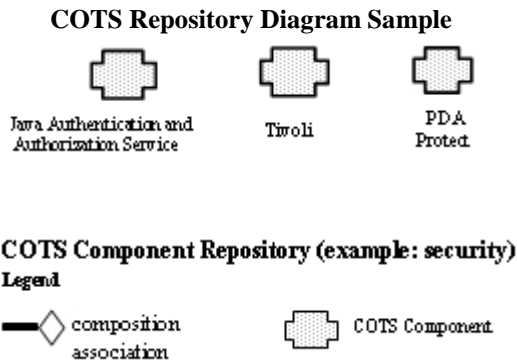
- display Impact specific icons in the menu and on the toolbar
- display Impact View in main browser
- capability to display, create, modify, and remove Impact diagrams
- capability to display and remove Agents
- capability to display and remove Goals
- capability to display and remove Use Cases
- capability to display and remove COTS Components
- capability to display and remove Associations
- capability to search for Agent, Goal, Use Case, and COTS objects
- store/retrieve Impact diagrams XML in ASCII file format
- store/retrieve Impact diagrams from RDBMS

Note. The impact diagram does not support defining or modifying elements.

2.2 COTS Component Repository

The COTS Component Repository allows for the definition and storage of COTS components. These definitions can then be mapped to functional requirements. The COTS Component Repository segregates the COTS Component data into four categories: Communication, Security, Multimedia, and Database.

Visual Model



Capabilities

Communication

- capability to display, create, modify, and remove IETF protocol diagrams
- capability to display, create, modify, and remove IEEE protocol diagrams

Security

- capability to display, create, modify, and remove Authentication diagrams
- capability to display, create, modify, and remove Authorization diagrams
- capability to display, create, modify, and remove Accounting diagrams
- capability to display, create, modify, and remove Privacy diagrams
- capability to display, create, modify, and remove Integrity diagrams

Multimedia

- capability to display, create, modify, and remove Audio diagrams
- capability to display, create, modify, and remove Video diagrams
- capability to display, create, modify, and remove Image diagrams

Database

- capability to display, create, modify, and remove Relational Database diagrams
- capability to display, create, modify, and remove Object Oriented Database diagrams

General

- display COTS Component Repository specific icons in the menu and on the toolbar (COTS component, generalization association, association, note, anchor)
- display COTS Component Repository View in main browser

- capability to display, create, modify, and remove COTS Component objects
- capability to display, create, modify, and remove COTS Component relationships
- capability to search for COTS Component objects
- capability to add, modify, remove Notes
- store/retrieve COTS Component Repository diagrams XML in ASCII file format
- store/retrieve COTS Component Repository diagrams from RDBMS

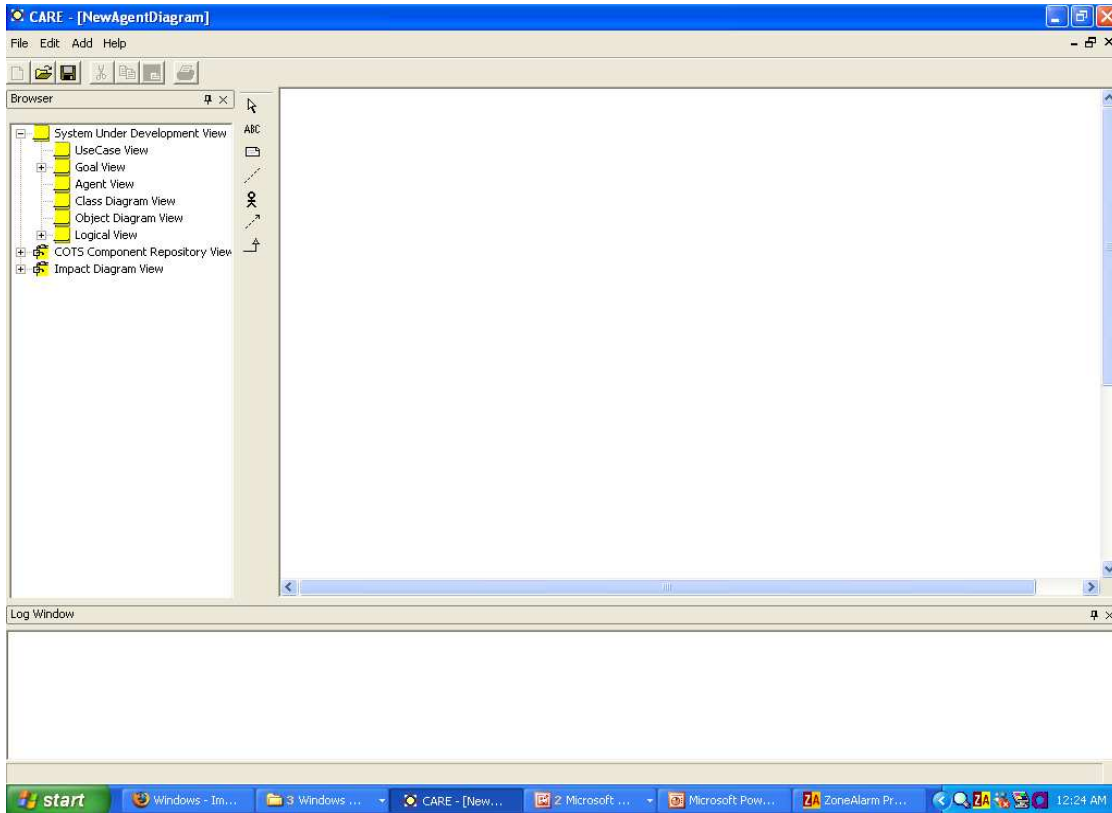
Specification

Name	Type	Cardinality
Identifier	string	1
Type	string	1
Name	string	1
Domain	string	1 or more
Keyword	string	1 or more
Overview	string	1
Vendor	vendor agent	1
Vendor Evaluation	string	1
Version Number	string	1
Operating Systems	nonfunctional requirement	1 or more
Interface type	nonfunctional requirement	1 or more
memory	nonfunctional requirement	0 or more
disk space	nonfunctional requirement	0 or more
Standards Compliance	nonfunctional requirement	0 or more
Performance	nonfunctional requirement	0 or more
Security	nonfunctional requirement	0 or more
Availability	nonfunctional requirement	0 or more
Reliability	nonfunctional requirement	0 or more
Scalability	nonfunctional requirement	0 or more
Subcomponents	COTS Component	0 or more
Assumption	string	0 or more
Notes	string	0 or more

3. Graphical User Interface

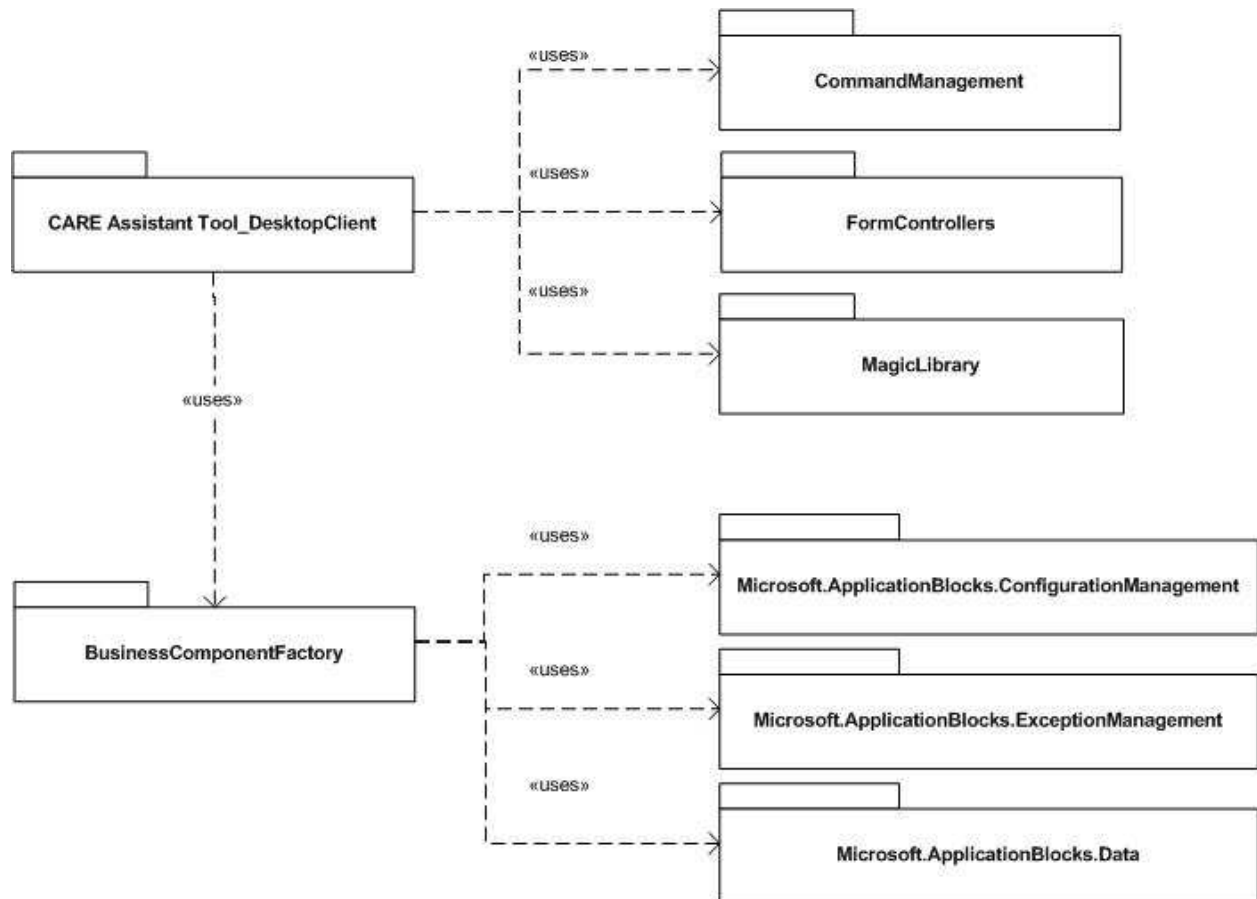
The overall layout of the CAT tool is planned to resemble standard CASE tools on the market today. This will provide a convenient, easy to use interface and minimize training time.

3.1 Main Form



4. Software Architecture

4.1 Logical View



where:

Desktop Client

The desktop client acts as a graphical user interface with the application user. Additionally, the desktop client interacts with all other system components and packages.

Command Management

This package provides the user interface with a mechanism for executing application commands.

Form Controllers

This package provides the user interface with a mechanism for interacting with the application forms.

Magic Library

This COTS package provides enhanced user interface capabilities.

Business Component Factory

This package is responsible for creating the business objects.

Microsoft.ApplicationBlocks.ConfigurationManagement

This COTS package provides automated capability to utilize XML based application configuration files.

Microsoft.ApplicationBlocks.ExceptionManagement

This COTS package provides enhanced exception management.

Microsoft.ApplicationBlocks.Data

This COTS package provides for interaction with a RDBMS.

4.2 Persistent Data

The CAT Assistant tool data is persisted on the file system of the client computer or a computer networked with the client computer. Currently, both the COTS Repository data and the diagram data are persisted to the file system as Simple Object Access Protocol (SOAP) message files.

SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics [2].

4.3 Concurrent Users

The CAT Assistant tool does not support concurrent users. This capability will be added in the next version of the tool.

4.4 Distributed Processing

The CAT Assistant tool does not support distributed processing. This capability will be added in the next version of the tool.

References

1. Cooper, K. Rampura, C. and Chung, L., "A COTS-Aware Requirements Engineering and Architecting Approach: Defining System Level Agents, Goals, Requirements and Architecture (Version 4), Technical Report UTDCS-24-05, Department of Computer Science, The University of Texas at Dallas
2. SOAP Version 1.2 Part 1: Messaging Framework W3C Recommendation 24 June 2003, available at: www.w3.org