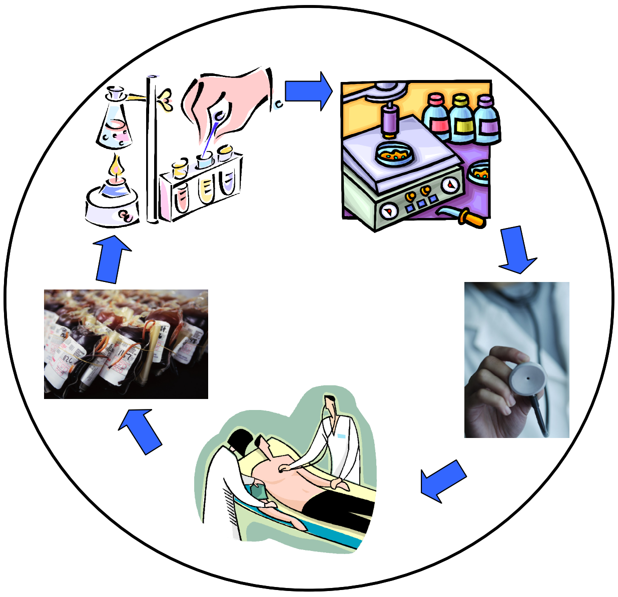
Patient-centred clinical validation using autonomous socialising knowledge agents



John McGrory, B.Sc., M.Sc., MBA.

Trinity College Dublin
Department of Computer Science
Faculty of Science
Dublin 2

A Thesis Submitted for the Degree of Doctor of Philosophy

March 2008

Head of Department: Professor Jane Grimson
Supervisor: Professor Jane Grimson

Declaration

I hereby declare that:

(a) This thesis has not been previously submitted as an exercise for a degree at this or any other University.

(b) This thesis is entirely the work of the author, except where otherwise stated.

(c) The Trinity College Library may lend and copy this thesis upon request.

(d) The copyright belongs jointly to the University of Dublin, Trinity College and John McGrory.

John McGrory
March 2008

Acknowledgements

I would like to thank everybody who has supported me whilst I have worked towards the completion of this thesis. However, I would like to extend a special thank you to the following people.

I would like to thank Prof. Jane Grimson for the support and guidance she has provided during the course of my research.

I would like to thank my parents and two brothers for the support and understanding they have shown.

I would especially like to thank my wife Sharon, daughter Rebecca and sons Sam and David. They have tolerantly sacrificed so much quality time so I could pursue this research, and during quality time they provided by stealth many analogies, such as the loss of a jigsaw box, which have assisted in describing design aspects of this thesis. I would not have been able to complete this work without that continual and absolute support.

Finally, I would like to thank Peter Gaffney, Chief Technologist in the Adelaide & Meath Hospital, Tallaght, Dublin and Frank Clark, lecturer in the School of Biological Sciences, Dublin Institute of Technology, Kevin Street, Dublin who continually encouraged the research, and offered their time and expertise freely to discuss the medical issues associated with this thesis.

I am greatly indebted to everyone concerned.

Publications

For publication purposes the underlying approach to patient-centred validation used in this thesis was divided into three strategic areas. The first was the examination of software agents so they could be used to represent clinical and laboratory guidelines. The second was the examination and comparison of medical and agent communications, so separate agents could communicate in compliance with medical standards. Finally, the third was the development of true patient centred validation using software agents. Each area was developed into a paper which has been accepted for inclusion in an International Conference on Health Informatics, HEALTHINF in January 2008.

McGrory John, Grimson Jane, Clarke Frank, Gaffney Peter, "*Agents representing medical guidelines*", presented at the HEALTHINF 2008 International Conference on Health informatics, Funchal, Madeira, Portugal, January 2008.

McGrory John, Grimson Jane, Clarke Frank, Gaffney Peter, "*Communication of medical information using agents*", presented at the HEALTHINF 2008 International Conference on Health informatics, Funchal, Madeira, Portugal, January 2008.

McGrory John, Grimson Jane, Clarke Frank, Gaffney Peter, "*True patient-centred laboratory validation using software agents*", presented at the HEALTHINF 2008 International Conference on Health informatics, Funchal, Madeira, Portugal, January 2008.

McGrory John, Grimson Jane, Clarke Frank, Gaffney Peter, "*The operation of a true patient-centred laboratory validation using software agents*", was accepted for presentation at the Artificial Intelligence and Applications AIA2008, 2008.

Abstract

Accurate patient-centred results generated from laboratory tests play a crucial role in the detection, diagnosis, and treatment of disease in a modern healthcare setting. The laboratory procedure which validates these results is developed using the contents of clinical and laboratory guidelines. These documents are standalone, self-contained bodies of knowledge which focus on a domain of interest such as a condition, organ or disease, and contain knowledge, aims, goals and procedures that generally encode best practice on managing (or understanding) that domain. Traditionally the knowledge and logic of these guidelines is decomposed into workflow activity paths and the motivation to achieve goals is managed centrally within a control engine. This means that all access to the encoded knowledge must be channelled through the centralised control engine, making distributed processing of the validation procedure difficult, and the provision of true patient-centred validation practically impossible. Patient-centred laboratory result validation is a procedure, which is customised around known patient information to ensure results are accurate, valid and plausible for that specific patient. This can only be achieved if the clinical or laboratory guideline is accessible to the parties performing the validation in its original integrated form. This thesis proposes that laboratory validation can be performed in a distributed and patient centred fashion using software agents, because guidelines have explicitly declared motivation components (in the form of desire and goals), which are linked by virtue of their overlapping viewpoints on a single domain. These key characteristics make it possible for an agent to act on behalf of a guideline without needing a centralised control engine, because the guideline contains motivation, knowledge and logic components. In addition to the elimination of a centralised control engine the presence of overlapping domain knowledge supports the collaboration activity between agents. The contribution made by this thesis is to advance the principle of patient-centred validation by developing and implementing a distributed network of *Autonomous Socialising Knowledge agents (ASK-agents),* which can act on behalf of a clinical or laboratory guideline, and collaborate using supportive messages in order to ensure results are accurate, valid and plausible for a specific patient.

Table of contents

At the back of this thesis a CD has been enclosed. This CD includes Appendices B, C, D and E.

Table of figures

Table of tables

Table of code examples

Chapter

1

Introduction

1.1 Introduction

A *Clinician* is a term used to describe doctors, physicians, nurses, therapists or other healthcare professionals who are licensed to deal with patients. Clinicians are medical experts who have, through training and experience, demonstrated proficiency in a particular area of medicine. Whenever a patient presents to a clinician for an appraisal, the clinician typically divides the meeting into three information gathering exercises [ASCLS, 2005]. Firstly, the clinician seeks relevant patient history, which they use to identify patient specific trends based on their health history or lifestyle. Secondly, the clinician examines the patient to identify their current status, for example listening to the heart and lungs using a stethoscope, measuring of blood pressure, observation, measuring patient tasks (e.g. movement, or hand-to-eye coordination) and touch. The third exercise is where the clinician orders tests to collect further supportive information to aid or confirm their diagnosis, and is only completed when the first two exercises provide insufficient information. From the point of view of this thesis and the Irish Medical Council, only doctors are permitted to order tests. But for consistency from an international publication standpoint, where persons other than doctors can perform this task, the term clinician is used. One of the most common forms of supportive information is through the reporting of laboratory test results, where patient fluid (i.e. blood, urine and mucus), cells, bacteria, parasites, micro-organisms or skin-biopsy samples are analysed and their results reported [Witte et al., 1997].

A laboratory result is an indicator or “snapshot” of patient health at the time of taking the sample, and is not considered in itself a diagnosis. However, these results play a crucial supporting role in the detection, diagnosis, treatment, and management of diseases in patients by clinicians in a modern healthcare setting [McLoughlin, 2006; Field, 1992]. These results have serious patient safety issues, because they are used to plan treatments, quantify medication amounts, and monitor patient responses [Plebani & Carraro, 1997]. So the most important activity undertaken by the laboratory is to ensure all reported results are *valid* and *plausible* for the patient from whom the sample was taken and is termed *result validation* [Westgard_d, 2002]. *Validity* indicates the degree of result correctness, and *plausibility* relates to the available evidence indicating a level of truth, but which may not always be enough to guarantee absolute certainty [Fraser_b, 2000].

The two main factors which can affect the *validity* and *plausibility* of laboratory results can be classified as analytical and non-analytical [Pierangelo et al., 2002; Fraser_b, 2000; Buttner et al., 1976]. Analytical factors primarily relate to the quality control activity that measures the exactness of the testing procedure and do not utilise any patient specific data. Non-analytical factors relate to patient preparation such as tourniquet application time and posture, and inherent biological variation which relates to an individuals' homeostatic set point, which is based on gender/age, biological cycle rhythm, and presence of disease/illness. Patient preparation is largely controlled through good working practices, staff training and standardisation of sample taking. An individual's homeostatic set point is a range around which a specific *analyte*[1] result from an individual varies over time [Fraser_b, 2000]. All laboratory tests that report specific analyte results are rated using an Index of Individuality (II), which is an indicator of expected result variation between different individuals on whom the testing procedure has been applied. If the II is large then the test is suitable for screening a large group of patients, however if the II is small, which is the case in the majority of laboratory tests (see section 2.4.2), results have a marked individuality, and are not suitable for screening. Although these non-analytical factors are largely outside the control of the laboratory, they remain a principal influencing issue affecting result plausibility. Therefore, the primary influencing factor in providing patient-centred validation is achieved through the management of the biological variation information, as the outcomes from the other factors are largely unaffected by patient specific data.

Biological variation information is predominantly disseminated to medical professionals through domain specific guidelines and protocols [Oosterhuis, 2004]. These clinical and laboratory guidelines and protocols are documents which focus on a domain of interest such as a condition, organ or disease, and contain knowledge, aims, goals and procedures that generally encode best practice on managing (or understanding) that domain. The content of the guidelines must be sufficiently precise to act as an aid to the reader, yet

[1] An *analyte* of a sample result is the constituent of the specimen to be measured.

remain general enough to be utilised by a large professional audience. Although the guidelines are integrated, standalone documents of expert knowledge, the information they hold can overlap and interweave in similar related domains of interest, by virtue of their use in medicine, for example hypertensive heart disease, coronary heart disease and hereditary heart disease. All of these diseases focus on the heart, but observe the organ from different viewpoints, which are established through various guidelines. It is the role of clinicians to care for specific patients, therefore clinicians and the laboratory staff who support them, must use these documents, in order to yield the maximum decision-making information based on the known facts about the individual.

The huge volume of samples being processed by the *clinical laboratory*[2], and the increasing time required to authenticate reports has resulted in a number of automated validation services being developed. These services correlate and manage guideline data and logic components, and orchestrate them in a fashion that can be applied to an individual patient's results [Peleg_d et al., 2003; Westgard_d, 2002; Rodney et al., 1998]. The most successful technique for automating clinical and laboratory guideline data and logic components is through Computer-Interpretable-Guidelines (CIG) [Kavanagh et al., 2002]. CIG's are software products that reproduce operational aspects of narrative guidelines to aid the clinicians and the laboratory technologists in ensuring that the available information and chosen healthcare paths are valid and appropriate for the patient. They manage this activity by converting the information and logic contained within the guideline into symbols that can be executed by a computer system [Peleg_c et al., 2004]. In CIG's, the knowledge and logic contained within the guideline is divided into a number of separate workflow paths and management rules which provide the motivation to achieve a task. Each workflow path represents a series of linked sub-activities. The management of the intersecting, merging and dividing of these linked activities to achieve a desire (or motivation) is achieved centrally within the CIG control switching management engine. Operational complexities relate to the technique of

[2] For example, Tallaght hospital which has 600-beds, approximately 21,000 in-patient discharges/year, 24,200 day cases/year and 204,700 out-patient appointments/year. The clinical laboratory processes on average 36,500 Biochemistry samples/month, 26,500 Haematology samples/month and 11,000 Microbiology samples/month. Each sample will have one or more tests performed on it, for example the 36,500 Biochemistry samples could equate to ≈ 290,500 separate analyte tests. The test-to-sample ratio varies depending on what information is sought by the clinician.

separating the guideline workflow paths and management rules. When another clinical or laboratory guideline is being added or removed from the CIG system, its logic and knowledge are inserted or extracted into/from the centralised engine, which manages the existing network of workflow paths. As a result, the management rules controlling the system become increasingly more complicated and *centralised*. This makes it difficult to distribute the rules for local processing to take place, which would permit workload sharing and introduce redundancy into the system to minimise the effect of downtime. When patient specific data is presented for validation, the CIG system processes information according to the guideline workflow paths and management rules, with minor customisation for different patients. However, the task of managing the maintenance and updating in order to avoid clashes between existing and new/altered activity, makes the system cumbersome and site specific [Kavanagh et al., 2002].

The fundamental issue is that clinical and laboratory guidelines are self-contained bodies of knowledge, which are consulted by medical professionals in light of known specific patient information, to ensure results are valid and plausible. The information contained in these documents act as a specific viewpoint understanding of a disease (or condition, or organ) of the human body, and there is some knowledge overlap between them. Consider, for example, clinical guidelines for liver failure, where each guideline is developed to give the reader a specific perspective of the disease from a particular viewpoint such as liver failure for patients with diabetes, or liver failure for renal failure patients. These guidelines specialise in their domain of interest and rarely deviate. The overlap between guidelines relates to partial guideline duplication which could be the understanding of an organ, or perhaps a procedure with different limits between adults and children, or domain specific concept. But the existing CIG approach alters the dynamic of guideline utilisation, by dividing the knowledge into a list of linked sub-activities. This change in operation from self-contained and autonomous bodies of knowledge to a flexible protocol makes it difficult to develop a patient-centred laboratory result validation service. This is because the existing automation system focuses on the general process of validation. For example, the selection of a particular activity path in the protocol is chosen based on patient characteristics such as age, gender or presence of another disease. These selection decisions are the *management rules* that tie the separate activity paths together. The encoded guideline knowledge does not retain its self-contained, integrated standalone capacity that can be accessed by other parties, to yield the maximum supportive information on behalf of a specific patient.

This thesis investigates the use of agents for accessing and managing guideline knowledge and logic. Agent-oriented architectures are used to reproduce aspects of human decision-making by combining attributes (beliefs), methods (plans) and desires (goals). The term *agent* is derived from the concept of *agency*, referring to employing somebody to act on one's behalf [Luck et al., 2004]. A *human-agent* represents a person, who interacts with others, acting on the person's behalf, to accomplish a predefined task [Turban_b et al., 2005]. A *software-agent* is a software module that represents a body of knowledge (i.e. application, person or instruction set) and interacts with others, acting on the body of knowledge's behalf, to accomplish a predefined task. To avoid confusion the term *human-agent* is used during analogy examples where the *human-agent* is a person such as a clinician, and the term *agent* relates to a *software-agent* throughout this thesis.

The principle of the agent operation is based on a *belief* capturing the informational attributes, the *desire* capturing motivational attributes and the *intention* capturing the deliberative attributes of an agent (known as BDI) [Rao et al., 1995], which when combined into a software application provides it with the capability to act autonomously and independently of any other module. The agent approach offers substantial advantages over current CIG approaches, because it retains the workflow paths and motivational management rules within a single autonomous module, without the need to have a centralised rule management engine. The agent's beliefs can capture and encode the information attributes of the guideline, the agent's plans can capture and encode the deliberative (i.e. choice between similar workflow paths) and action attributes of the guideline, and the desire can capture and encode the motivational attributes of the guideline. This allows the agent to act faithfully and autonomously on behalf of the guideline in a self-contained capacity [Pokahr_b et al., 2005; Luck et al., 2004]. When patient specific information is presented to the individual agents, they have the ability to apply their encoded knowledge and logic, and provide a supportive response based solely on that information. Using this approach an agent module can yield the maximum supportive response from the other agent. By developing an agent and environment that supports separate agents broadcasting supportive communications to each other, the agent approach offers the opportunity for the laboratory results to be validated in a patient-centred fashion.

The aim of this thesis is to develop a laboratory validation service, which supports separate autonomous agents, each acting on behalf of a body of self-contained knowledge (i.e. clinical and laboratory guidelines) to collaborate with each other in order to ensure results are valid and plausible for the specific patient. The term given in this thesis to these types of agents is *Autonomous Socialising Knowledge agents* (ASK-agents).

1.2 Motivations

The work presented in this thesis demonstrates that the agent-orientated paradigm represents a useful and novel approach to managing the complexities, which are inherent within systems that capture and utilise independent, self-contained bodies of knowledge, such as clinical and laboratory guidelines. The rationale behind proposing the agent architecture is based on the premise that if the logic, knowledge, and motivational aspects of an independent, self-contained body of knowledge can be reproduced by the BDI of an agent, it would permit the agent knowledge to be utilised and accessed directly as a self-contained entity. Messages could be transmitted to the ASK-agent, or beliefs updated based on the agents' environment and processed through its BDI. This action then produces appropriate agent activity, to yield the maximum supportive response outputs from the ASK-agents' interpretation of the guidelines based on the known facts about the individual patient. This dynamic is more comparable to the true use of clinical and laboratory guidelines than current encoding implementations. Operating solely using their influencing algorithm, ASK-agents can act autonomously in their area of expertise, or collaborate with other ASK-agents, or other types of agents in social communities. The task of managing the maintenance and updating of the ASK-agent system, as the guidelines themselves are reviewed, is easier, because a single agent acts on behalf of a single guideline. The need to manage a centralised network of management rules to control the intersecting, merging and dividing of these workflow paths is mainly avoided, and the processing can be distributed around a network with ease. In addition to the freedom to provide distributed validation processing, the validation procedures for patient data becomes site independent. This is due to the autonomous interaction of the ASK-agents as a social community. A site independent technique can substantially reduce the development and running costs due to the capability to reuse the application on many laboratory sites.

The ASK-agent dynamic also provides the possibility for result validation to be completed using a variety of alternative validation measurements and procedures that can be accessed at run-time, instead of being established at compile-time. This facilitates the ASK-agent approach to yield the maximum supportive response outputs from all available ASK-agents representing guidelines.

1.3 Objectives

The key objective of this research is to develop an ASK-agent that can capture and represent a body of self-contained independent knowledge, and permit it to be accessed and utilised by other systems as a single autonomous entity. For the purposes of this research the form of self-contained independent knowledge used will be clinical and laboratory guidelines. Currently agent platforms do not possess the ability to convert clinical and laboratory guidelines into autonomous software modules, and do not have the communication capability to permit these entities to socialise in order to provide patient-centred validation. Therefore, the main aims of the research are to:

1. Validate that BDI based agents, with the addition of the novel ASK-agent components, can accurately reproduce the logic and knowledge contained in clinical and laboratory guidelines. This includes producing a set of functions, which can represent the content of guidelines preserving the original motivational aspects.

2. Develop an environment that allows two or more ASK-agents to collaborate on linked activities. This collaboration will use supportive response message communications, which are generated based on each agent's interpretation of the patient specific data.

3. Develop an ASK-agent shell, which allows the logic and knowledge representation contained therein to be accessed and utilised by other software applications.

4. Using the ASK-agent approach, develop a validation method that supports clinicians and laboratory technologists in providing patient-centred healthcare delivery.

5. Develop a technique to permit ASK-agents to be added, deleted or altered without system reconfiguration.

6. Develop a technique to improve the Index of Individuality (II) for groups of laboratory tests, by allowing ASK-agents to share supportive information to aid result validation.

7. Evaluate the operation of the developed ASK-agent approach in the validation of laboratory results within a clinical laboratory setting.

Contributions made by this research primarily focus on two areas. Firstly, the use of agents as an instrument in medical informatics is advanced by the development of the ASK-agent, which progresses base-level agent operations and functions making them medical software compliant. This permits the agency concept to be utilised within the automation of medical information. Secondly, the development of the ASK-agent advances patient-centred validation by developing a novel process dynamic, which matches the true intended use of clinical and laboratory guidelines, and provides a prototype application to demonstrate the concept.

Therefore, in achieving these aims the thesis makes the following contributions:

1. Creates a novel innovative approach, which utilises the sharing of supportive information between autonomous self-contained software entities that have overlapping knowledge, without the need for a centralised element of control to manage the interfaces between these resources.

2. Advances agent design by improving their ability to collaborate through the addition of a social fabric, so separate agents can collaborate on related activities.

3. Creates an ASK-agent-to-guideline information map which can accurately preserve the original intention (motivation), knowledge and logic of guidelines and protocols.

4. Provides a system that supports the adding, deleting or altering of the autonomous ASK-agents modules without reconfiguring the system.

5. Provides a deeper understanding of how information, motivation, knowledge and logic contained in standalone documents can be better described by the authors in order that their content can be effectively computerised.

1.4 Thesis Outline

To achieve the objectives outlined above, the remainder of this thesis has been organised as follows:

Chapter 2: Overview of medical laboratory science. This chapter examines medical laboratory science from the *patient-centred* result validation viewpoint. The chapter details the validation cycle and covers areas such as drug and food interaction/interference, reference models, medical information communication standards, units of measurement, guidelines and protocols. This chapter focuses on the particular aspect of the validation cycle which is most affected by patient information. This chapter also presents an overview of liver disease, the medical domain used in the proof of concept.

Chapter 3: Overview of agent software and collaboration software. This chapter presents an overview of the state of the art of agent and collaborating software in the context of patient-centred validation. The chapter covers networking, communications, knowledge source representation, computer based reasoning, agent software architecture, software modelling, collaborative computer systems, ontologies and languages.

Chapter 4: Patient-centred validation service analysis. This chapter analyses aspects of computing, information and communication technologies that can be effectively applied to patient-centred validation. It presents the characteristics of the system and a functional requirements specification for the base-level platform, from which an agent platform was chosen and the extended functions developed.

Chapter 5: Patient-centred validation service design. This chapter progresses the architectures and methodologies analysed in chapter 4 by detailing the ASK-agent architecture. The chapter then continues by illustrating an example of the procedures used to convert clinical and laboratory guidelines into separate ASK-agent modules. This procedure divides ASK-agent development into logic steps, some of which expand on existing procedures specifically the Arden Syntax Medical Logic Modules (MLM).

Chapter 6: Patient-centred validation service implementation. This chapter develops an implementation of the ASK-agent architecture in order to demonstrate its operation and performance. The implementation is based on the realistic problem of evaluating the source of anaemia of chronic diseases using liver function tests.

Chapter 7: Results and discussion. This chapter observes the output from the proof of concept implementation based on a specific set of patient information. It discusses the results obtained from the implementation based on a set of synthetic test data. The chapter continues by predicting the network traffic overhead and how management and expert groups can be established. Finally the chapter discusses the contribution to knowledge the ASK-agent architecture makes.

Chapter 8: Conclusions and recommendations. This concluding chapter examines the new ASK-agent architecture in relation to the research objective. The chapter then targets areas where future research could be completed.

Chapter

2

Overview of medical laboratory science

2.1 Introduction

The aim of this chapter is to provide an overview of medical laboratory science and particularly to illustrate the complexities involved, and factors affecting outcomes, when providing a laboratory result validation service. It provides a medical background in order to produce a requirements specification for the ASK-agent architecture. Section 2.2 introduces patient-centred validation, and discusses why it is vital for a laboratory to customise its procedures to validate results with respect to an individual patient [Collier, 2001; Fraser_a, 2000].

In order to appreciate the complexities associated with validating results, it is important to view the different operations and procedures performed by the laboratory within the context of the entire validation cycle, from the point of where samples enter the laboratory through to when the results are reported. Section 2.3 provides an overview of the laboratory validation cycle where the key stages are identified and discussed. The most important activity performed by the clinical laboratory is to ensure the validity and plausibility of the result for the specific patient [Westgard_d, 2002; Fraser_b, 2000]. Section 2.4 examines the procedures used by the laboratory to ensure the validity and plausibility of the result.

The operations and procedures used by both the laboratory and clinicians are aided by guidelines and protocols. Clinical and laboratory guidelines and protocols are the main channel by which domain specific aims, goals and best practice information are disseminated to medical professionals [Oosterhuis 2004; Overhage et al., 1997]. Other dissemination channels include web sources, internships and published literature such as text books, journals, papers and brochures. The purpose of a guideline is to aid clinicians by following best practice activities according to accepted, recognised and standardised procedures [Ellrodt et al., 1995]. Section 2.5 examines the role that guidelines and protocols play in the design and operation of laboratory validation systems.

The ability of a laboratory to provide a validation service is supported to a large extent by a group of software applications collectively known as a Laboratory Information System (LIS). A LIS is used by laboratory staff to manage and coordinate their day to day activities [Burtis et al., 1998]. It provides facilities to store patient information (i.e. ID, date of birth, weight, height, gender etc), quality control information (i.e. control samples, calibration etc.) and create reports. The LIS can also manage workflows and provide an ability to interpret results. Section 2.6 examines the role of the LIS in validating results, and discusses the main components underpinning its operation.

In order to ensure that the ASK-agent can accurately validate laboratory results, it is essential that the domain used for demonstration is carefully chosen. The medical domain must provide an opportunity for the system to show its ability to handle the most important features used in laboratory validation. Section 2.7 introduces the medical domain, which will be used during the development of the proof of concept implementation.

Finally section 2.8 presents a discussion of the factors affecting the reported result of a patient-centred laboratory validation service.

2.2 Patient-centred validation service

The concept of '*patient-centred'* laboratory result validation is best described using an example of a patient seeking customised healthcare. Although patient information is stored in their healthcare record, the information pertaining to a patient's healthcare delivery and management plans is distributed in various configurations and levels of abstraction (i.e. legislation, guidelines, best practice documentations, local government policy, institute policy, World Health Organisation (WHO), etc.). Each of these healthcare delivery and management instruments is focused on a specific area (i.e. medical entitlements, social services, subsidies, financial entitlements), but are not presented in a consistent format as the viewpoints, purposes, and backgrounds of the authors are different in most cases. To determine their personalised healthcare privileges, entitlements and care, the patient or their agent (clinician) needs to sift through the various documents. This procedure of screening expert information and knowledge, in some cases completed mentally by the clinician when guideline information is remembered, on behalf of the patient is deemed '*patient-centred'*. Therefore, '*patient-centred*' laboratory result validation is a *quality control* course of action used to shape, or customise result validation procedures around known patient information to ensure results are accurate, valid, and plausible [Van Bemmel et al., 2002]. The standard validation service in use by laboratories on the other hand, performs a generic validation process on all patients equally. The patient's information is stored in their healthcare record and the validation procedure uses this patient information to navigate down along the generic validation decision rule set. All patient results start their validation path at the same point and a decision at each step is made based on the available patient information (e.g. age, gender, height, weight). This validation is carried out irrespective of any additional (mitigating) specific information or data known about the patient, which may fall outside the generic rule set. The additional information would be contained elsewhere within a guideline but omitted for rule construction simplicity, where the patient information is at various levels of decision making and a general decision is not possible to cover all avenues. In these cases it is common practice that the most likely avenue is encoded, and the less likely omitted. The '*patient-centred*' approach has an increased level of operational complexity when compared to the standardised validation service [Little et al., 2001], due to the larger number of result validation permutations available for patients within a medical institution. But the result gained in providing this service improves the quality of patient care through accurate and specific patient information [Stewart, 2006].

2.3 Overview of the laboratory validation cycle

The purpose of a laboratory test is to provide supportive information for the clinician. So when a test is ordered the sample taking procedure is scheduled, the sample (e.g. blood, urine, mucus, saliva) is taken, and then the sample is sent to the clinical laboratory for analysis. The laboratory analyses the sample and produces a set of results. Before issuing these results to the clinician, the laboratory staff must ensure the results are *valid* and *plausible* for the specific patient from whom the sample was taken [Westgard_d, 2002]. An illustration of the patient sample laboratory cycle is shown in Figure 2.1 followed by an explanation of the cycle.

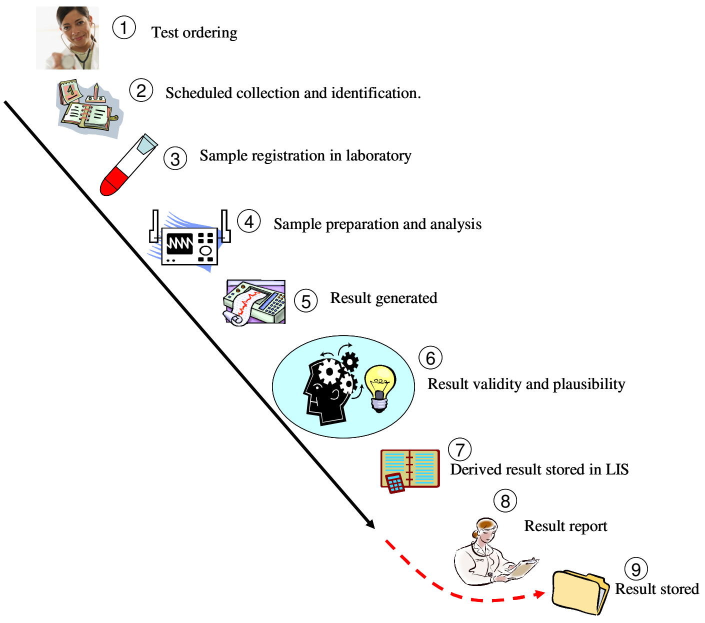

1 Test ordering
2 Scheduled collection and identification.
3 Sample registration in laboratory
4 Sample preparation and analysis
5 Result generated
6 Result validity and plausibility
7 Derived result stored in LIS
8 Result report
9 Result stored

Figure 2.1 Patient sample validation cycle

Figure 2.1 Patient sample validation cycle explanation:

(1) **Test ordering**

The clinician fills in a sample request order.

(2) **Scheduled collection and identification**

Depending on the type of sample, the procedure to take it is scheduled. In some cases the patient needs to fast or be prepared (e.g. surgery) before the sample can be taken. When the sample is taken it is placed within a handling container (some of which contain sample preservation chemicals) and given a unique identification marking (e.g. barcode) so it can be traced back using the LIS to the patient, doctor and sample type.

(3) **Sample registration in laboratory**

From the moment a sample (e.g. blood, urine) leaves the patient's body it begins to change (e.g. oxygen levels decrease). So when the sample enters the laboratory the date, time and identification markings are logged in the LIS in addition to the labelled information. Similar sample types requiring the same analysis are grouped together into *batches* for easier identification, handling and processing. The processing of a set of samples in a batch is termed a *batch run*. For quality control purposes a batch of patient samples will also include a number of control samples. These control samples have a similar composition (also known as a matrix) to the sample, but their analyte concentration is known before the batch is analysed.

(4) **Sample preparation and analysis**

Some samples are pre-treated, if necessary, prior to testing to enhance their accuracy. The samples are then loaded into the analyser and examined (e.g. as spectrometer, chemical reaction, fluorescence under different wavelengths of light, electrical current etc.).

(5) **Result generated**

On completion of the analysis, the testing procedure produces an output in the form of a comment or numerical measurement. Modern clinical analyser results can be printed out, displayed locally on the analyser, or networked and saved in the LIS.

(6) **Result validity and plausibility**

The laboratory must determine with certainty that the generated result is *valid* and *plausible* for the patient, and these activities are examined in section 2.4.

(7) **Derived result stored in LIS**

On completion of the validation procedure, if the result remains plausible, it becomes a *derived result,* and is stored in a section of the LIS that is accessible to clinicians. Up until this point results stored in the LIS are only available to the laboratory staff and used for retesting comparison purposes, or to correlate with another type of patient sample.

(8) **Result report**

The laboratory manager returns the result outcome to the *clinician* who ordered the test. In some institutions this can be completed online.

(9) **Result stored**

The clinician receiving the result uses it to aid in their patient-centred healthcare delivery, and enters the result into the patient's healthcare record.

2.4 Result validity and plausibility

There are two principal factors affecting the *validity and plausibility*[3] of the result and they are: analytical or non-analytical [Fraser_a, 2000; Buttner et al., 1976]. Sections 2.4.1 and 2.4.2 discuss the analytical and non-analytical factors respectively. In addition to these distinct factors section 2.4.3 introduces other generic aspects which complicate laboratory validation from a laboratory testing and Information Technology (IT) perspective.

2.4.1 Analytical factors

Analytical factors can be sub-divided into two groups relating to the generated result: *precision and accuracy* aspects, and *sensitivity and specificity* aspects. On the whole the processing of these factors is strongly linked to testing, quality control and statistical analysis.

Precision and accuracy aspects

The *precision and accuracy* of results are an indication of a testing procedure's *exactness*. This *exactness* is realised by including control samples in the batch runs, and comparing the reported result for these control samples to their known value. The term *accuracy* relates to the degree of authenticity of the measured value when compared to a reference value, while *precision* is the degree of reproducibility of the measured value around the *accuracy* result [Fraser_b, 2000; Bachmann, 2000]. The measurement of precision and accuracy is a process based on the statistical collection and analysis of data, such as analyte results [Chatfield, 1983]. The main concept is to identify and quantify the underlying important characteristics of a data set in order to understand and interpret them. A normal (Gaussian) distribution graph that is used to interpret control sample results is shown in Figure 2.2, and illustrates graphically the terms accuracy and precision.

[3] *Validity* indicates the degree of result correctness, and *plausibility* relates to the available evidence indicating a level of truth, but which may not always be enough to guarantee absolute certainty [Fraser, 2000].

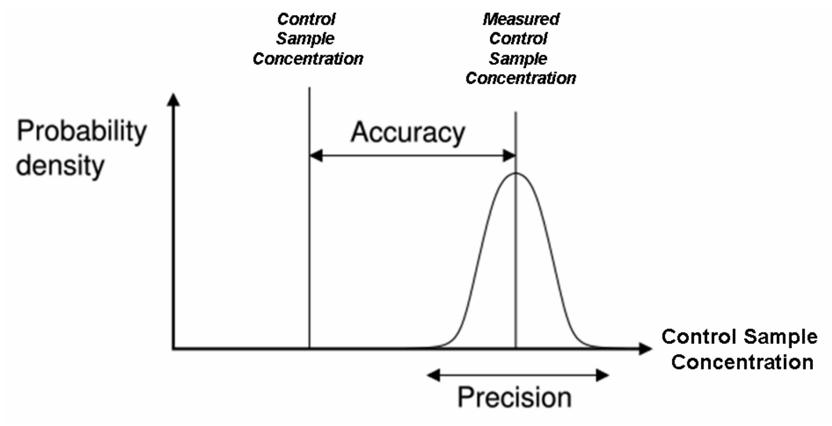

Control Sample Concentration
Measured Control Sample Concentration
Probability density
Accuracy
Control Sample Concentration
Precision

Figure 2.2 Precision and accuracy of control sample data

The standard deviation of a set of test results is used as part of the statistical analysis of samples, and can often change with sample concentration [Westgard_e, 2005; Fraser_b, 2000]. Sample concentration is a measurement of units per volume. From a laboratory standpoint a result must be produced without the actual concentration being a mitigating factor, because there is a large variation in sample concentrations between individuals. The coefficient of variation (CV) describes the standard deviation as a percentage of the mean, and is calculated using equation (2.1). It reflects the CV as a ratio of the standard deviation to the concentration. So it often provides a better estimate of a testing method's performance for a range of concentrations, and the concept can be reused to obtain important statistical data with regard to different types of laboratory tests, individuals and groups.

$\bar{x}$	=	Mean
SD	=	Standard Deviation

$$CV = \frac{SD}{\bar{x}} \times 100 \qquad (2.1)$$

An example of this reuse is when the CV equation is employed to identify inherent variations in analytical systems, such as the *analytical coefficient of variation* (CV_A). The CV_A quantifies changeability of the result due to fluctuations in temperature, variability in sample volume, reagent delivered, changes in the environment, and inconsistent handling of materials [Westgard_e, 2005; NHLBI, 2004]. The CV_A can be reduced in a laboratory when test sample duplicates are performed, where the same test is completed more than once under different conditions (e.g. temperature, sample volume etc.) using identical control samples, and comparing the results over days weeks and months using equation (2.1).

Sensitivity and specificity aspects

The choice of testing procedure used for a particular sample plays a crucial role in the analytical *sensitivity* and *specificity* of the result [Fraser_b, 2000; Bachmann, 2000]. Test s*ensitivity* is defined as the ratio of true positive results to the total number of affected (positive) patients tested, expressed as a percentage [Westgard_e, 2005; Fraser_b, 2000]. To clarify, consider the example of a biopsy result for cancer. A *positive* result indicates cancer present, whereas a *negative* indicates no cancer present. A *false negative* is where a result is returned as *negative*, but the sample actually contains cancer. A *false positive* is where the result returned is *positive,* but actually contains no cancer. Therefore, test s*ensitivity* measures a test's ability to correctly identify the presence of a disease. A test with high *sensitivity* has a small number of *false negatives*.

Test s*pecificity* is defined as the ratio of true negative tests to the total number of unaffected patients tested, expressed as a percentage. It measures a test's ability to correctly identify the absence of disease [Westgard_e, 2005; Fraser_b, 2000]. A test with high *specificity* has a small number of *false positives*. Both test *sensitivity* and *specificity* are independent of disease prevalence in the community. Whilst choosing a laboratory test it is important to consider the factors that would affect the *sensitivity* and *specificity* of a test when being applied to a cohort of the population. Some factors affecting the choice of test are: *pre-treatment, cost, matrix, same batch control material, operator error, precision of figures, testing range, drug and food interaction, and SI units* [Fraser_b, 2000; Westgard_a, 1997; Westgard_d, 2002].

In summary, the analytical factor components such as precision, accuracy, sensitivity, and specificity at no point refer to patient specific data, and therefore are not patient-centred components, but more specifically laboratory procedure centred.

2.4.2 Non-analytical factors

Non-analytical factors affecting clinical laboratory result outcomes relate principally to pre-analytical variation, and inherent biological variation. Pre-analytical variation is associated with the preparation of the individual before and during the sample taking, and with the sample after it has been taken, such as posture, tourniquet application time, duration of sample taking, transportation time, storage and type of sample [NHLBI, 2004]. This factor is predominantly controlled through good working practices and through healthcare professional training, but does not involve any patient specific data, and therefore in relation to this research is assumed to be in place and working effectively. Biological variation on the other hand relates to the deviation of an individual's analyte result around the patient's *homeostatic set point*[4]. Contrary to all other factors and aspects, biological variation does utilise, and depends on, patient specific data in order to ensure result validity and plausibility. The remainder of this section is divided under two headings to aid in identifying, characterising and discussing patient specific data in subsequent chapters and they are; biological variation and index of individuality.

Biological variation

If an identical testing procedure was performed on four individuals, it would be observed that a specific analyte measured in samples from each person would never return exactly the same result. If the same test was performed on an individual over time (e.g. one month), it would be observed that the individual's homeostatic set-point of the specific analyte varies in a random manner [Collier, 2001; Fraser_b, 2000; Burtis et al., 1998; Marshall et al., 1995]. This fluctuation is due to three factors: pre-analytical variation, analytical coefficient of variation (CV_A) and inherent biological variation [Fraser_b, 2000]. It was already shown that the pre-analytical variation, analytical coefficient of variation (CV_A) does not utilise patient specific data. The only part of the fluctuation that relates to patient specific data is the inherent biological variation. Biological variation can be classed as inter-individual, or intra-individual. When a homeostatic difference is

[4] *Homeostatic set point* is a range around which a specific analyte result from an individual varies over time.

applied between individuals it is referred to as inter-individual biological variation (or between-subject biological variation - CV_G) [NHLBI, 2004; Fraser_b, 2000]. When the difference applies within an individual it is referred to as intra-individual biological variation (or within-subject biological variation - CV_I). The following example taken from Fraser_b illustrates the biological variation classes [Fraser_b, 2000]. Table 2.1 shows the serum Creatinine concentrations [*µmol/L*[5]], and Table 2.2 shows the serum Iron concentrations [µmol/L] from four healthy individuals, in six samples taken at daily intervals with their respective personal mean, lower range and upper range values.

A measurement of the serum Creatinine level is often used to evaluate kidney function. The normal reference range for serum Creatinine is between 53-106 µmol/l for men and 44-80 µmol/l for women. The Creatinine level can also be detected in the urine and the normal reference range for urine Creatinine is between 8.84-13.3 mmol/24 hours for both men and women. The term *reference range* relates to the observed upper and lower limits of an analyte in a particular reference population cohort. It is not always necessary that the range is based on a healthy population; in fact some ranges are used to identify specific illnesses in patients. Individuals or a group can be compared to the upper and lower limits of the chosen cohort to assess or determine their health status. The Iron test is used to measure the amount of Iron carried by transferrin, which transports Iron from the gut to cells that use it [Burtis et al., 1998]. In people with *anaemia*[6], the Iron test can help to tell whether the anaemia is due to Iron deficiency, but can also help to recognise when *anaemia* is normal. The normal range for serum Iron is between 10-26 µmol/l for both men and women.

[5] The term *mol* is an SI base unit that measures an amount of substance. One mol contains Avogadro's constant (approximately 6.022×10^{23}) entities. The term *mol/L* indicates the amount of the substance to a volume of one liter. The *µ* or *m* prefix dictates the engineering units to be applied to the amount of the substance.

[6] *Anaemia* is the most common disorder of the blood and is described as a deficiency of red blood cells and/or haemoglobin in the blood [NAAC, 2005; Kumar, 2002]. In most cases it is indicative of a chronic illness or cancer, but in certain cases *anaemia* is normal due to the metabolism of the patient.

Description	**Subject 1**	**Subject 2**	**Subject 3**	**Subject 4**
Sample No 1	86	124	97	144
Sample No 2	81	128	93	139
Sample No 3	82	120	95	141
Sample No 4	84	126	91	138
Sample No 5	80	130	97	146
Sample No 6	85	128	91	144
Mean	83	126	94	142
Lowest Value	81	120	91	138
Highest Value	85	130	97	146

Table 2.1 Serum Creatinine [μmol/L] from four healthy individuals [Fraser_b, 2000]

Description	**Subject 1**	**Subject 2**	**Subject 3**	**Subject 4**
Sample No 1	8	19	11	10
Sample No 2	18	9	16	25
Sample No 3	14	23	23	15
Sample No 4	22	17	19	20
Sample No 5	10	20	27	14
Sample No 6	16	27	21	18
Mean	16	19.2	21.2	18.4
Lowest Value	8	9	11	10
Highest Value	22	27	27	25

Table 2.2 Serum Iron [μmol/L] from four healthy individuals [Fraser_b, 2000]

The Creatinine and Iron tables are illustrated graphically in Figure 2.3 and Figure 2.4 respectively, complete with the upper and lower normal reference limits, indicated by the two horizontal broken lines. The vertical line for each individual refers to the range between the highest and lowest value, and the short horizontal line indicates the mean. Figure 2.3 for serum Creatinine illustrates personal ranges for individual's #1 and #3 never exceed, whereas those for #2 and #4 never fall within the normal reference range, even though the individuals are healthy. Therefore, a validation technique for a patient's Serum Creatinine analyte that relies solely on normal reference ranges will be unable to determine if individuals #2 and #4 are in good or poor health. It is important to emphasise that it does not signify that the normal reference ranges are wrong. It simply indicates that individuals #2 and #4 do not fall within the mass population normal range category for this specific analyte, and that the inter-individuals' ranges vary extensively making generalised limits difficult to set.

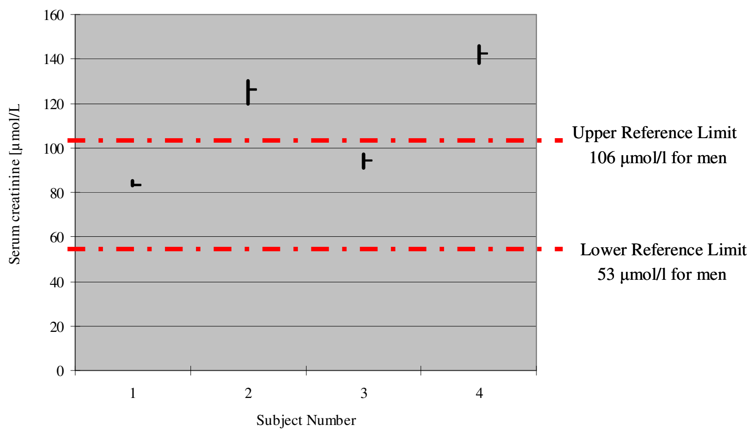


Figure 2.3 Variations between-subject for serum Creatinine [µmol/L]

Figure 2.4 for serum Iron conversely illustrates that the majority of the four individual personal results fall within the normal reference limits, and only the extreme personal high or low results exceed them. Testing procedures generating these types of results makes it considerably easier to determine the patient's status.

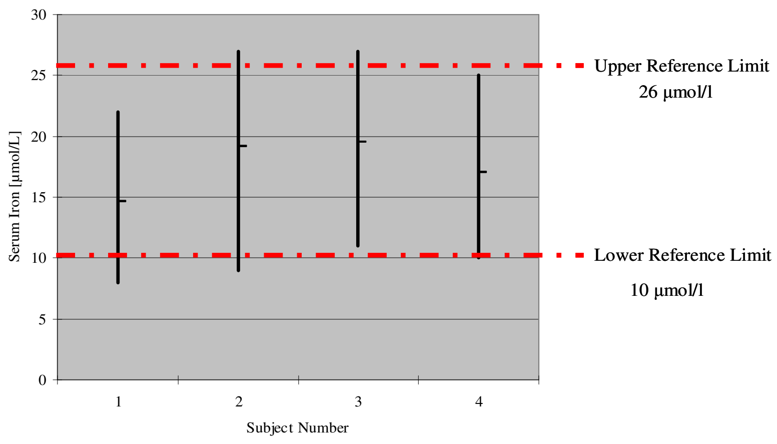


Figure 2.4 Variations between-subject for serum Iron [µmol/L]

Within-subject biological variation CV_I expresses the standard deviation of an individual's results as a percentage of the individual's mean. The CV_I is an indication of the variation spread in an individual's results. The formula to calculate the CV_I is given in expression (2.2) and based on the CV equation given earlier in this chapter as equation (2.1):

$\bar{x}$	=	An individual's mean result.
SD	=	An individual's standard deviation result.

$$CV_I = \frac{SD}{\bar{x}} \times 100 \qquad (2.2)$$

The between-subject biological variation CV_G expresses the standard deviation of a group of individuals' results as a percentage of the group mean. The CV_G is an indication of the variation spread in a group of individual's results. The formula to calculate the CV_G is given in expression (2.3):

$\bar{x}$	=	A group of individuals mean result.
SD	=	A group of individuals standard deviation result.

$$CV_G = \frac{SD}{\bar{x}} \times 100 \qquad (2.3)$$

If the calculated within-subject biological variation CV_I is dramatically smaller than the between-subject biological variation CV_G (i.e. $CV_I << CV_G$) the individual being measured using a specific testing procedure is said to have a *marked individuality* [Fraser_b, 2000]. However, if the within-subject biological variation CV_I is similar to the between-subject biological variation CV_G (i.e. $CV_I \approx CV_G$) the individual being measured using a specific testing procedure is said to have *little individuality.*

If two or more recent results have been processed by the laboratory the variation between consecutive pairs can be calculated (known as the '*delta value')*, and used by the laboratory technologists to facilitate the validation procedure. The fundamental basis of delta-checking is, if a patient is stable the '*delta value'* should be small, but if the '*delta value*' is greater than a pre-defined limit (as set by guidelines), it is an indication that there has been a real change in the patient's status, or that there are errors associated with result processing [Fraser_b, 2000; Lacher, 1990]. The defining of appropriate reference '*delta value'* variations must take into account the CV_I.

Index of Individuality

An analyte's individuality itself can be measured when using a specific testing procedure, and is termed the *index of individuality* (II) [Collier, 2001; Fraser_b, 2000]. The formula to calculate the *index of individuality* is given in equation (2.4).

$$II = \frac{\sqrt{\left[CV^2{}_A + CV^2{}_I\right]}}{CV_G} \tag{2.4}$$

Where CV_A is the analytical coefficient of variation.

A low index of individuality means the analyte has a marked individuality, and conversely a high index of individuality means the analyte has little individuality. With an *index of individuality* <0.6 the normal reference limits will be of little utility, especially in identifying a significant change in a result or in a patient. This is illustrated with the between-subject Creatinine range chart shown in Figure 2.3, where the group of individual's ranges rarely overlap. The within-subject Creatinine range is smaller than the between-subject variation, making the resulting *index of individuality* small [Collier, 2001; Fraser_b, 2000]. Consider for example, individual #3 in Figure 2.3, where their Creatinine results range from a personal upper and lower range that never exceeds the lower or upper normal reference limits of the test. Therefore, if the patient was ill and the Creatinine value changed, the population based reference range on its own would be incapable of indicating any early warning to the laboratory technologists. The presence of an appropriate delta value increases the chances of the laboratory technologists being alerted to the patient's results, because the population based reference limits were not reached. On the other hand laboratory tests with the *index of individuality* >1.4, have normal reference limits that are capable of indicating an early warning for all individuals, refer to the within-subject Iron chart shown in Figure 2.4, where the ranges clearly overlap. The subject-iron range approximately matches the between-subject variation, making the resulting *index of individuality* large.

Laboratory technologists validate tests with a high *index of individuality* by comparing the results to the population based reference range (i.e. normal reference limits). If the result exceeds the population based reference range limits then the laboratory can question the plausibility of the result. In some cases the technologist can perform another type of test just to provide some further supporting evidence with regard to the sample result. But how many laboratory tests have a high *index of individuality?* Table 2.3 identifies a number of tests performed by the clinical laboratory, and places them in order of their *index of individuality.* The (U) suffix after the laboratory test name relates to analysis which was performed on urine based samples. Laboratory tests with an *index of individuality* <0.4 are listed on the left of the table, while laboratory tests with an *index of individuality* >1.4 are listed on the right, and all other tests in between are grouped using *index of individuality* values from 0.4 to 1.4. This table shows that the majority of all clinical laboratory tests fall into the *index of individuality* <1.4 threshold.

This presents the laboratory technologists with a further validation conundrum! How can technologists validate sample results in a patient-centred way, if the majority of tests have an *index of individuality* <1.4? In some tests the *index of individuality* is improved by developing a pattern of predictable cyclical rhythms, which may be daily, monthly, or seasonal in nature, or grouping or subdividing (stratification) the cohort of tested individuals. The majority of this type of information is disseminated to medical professionals using guidelines and protocols. Knowledge of rhythm cycles is important, because it would be very difficult to create population-based reference values for each and every point in a cycle [Fraser_b, 2000]. This means appropriate times to investigate a cycle must be defined, and reference values for these cycle episodes generated. Moreover, the absence of predicted rhythms may indicate the presence of disease. Stratification on the other hand involves dividing a sample into homogeneous sub-samples based on certain characteristics of the population.

Index of Individuality

<0.4	0.4 - 0.59	0.6 - 0.99	1.0 - 1.4	>1.4
<-------------->	<-------------->	<-------------->	<-------------->	<--------------->
PSA	Creatinine	Potassium	Sodium	Lactate
CEA	Uric Acid	Calcium	Chloride	pH
LDL	Cholesterol	Magnesium	Calcium, Ionized	Protein (U)
Prolactin	HDL	Glucose	Phosphate	Sodium (U)
FSH	Triglycerides	Urea	pCO2	
B2Microglobulin	Myoglobin	Bilirubin, Conjugated	Osmolality	
Homocysteine	LH	Bilirubin, Total	Ferritin	
ALP	ALT	AST	Iron	
CKMB	CK	LD	Potassium (U)	
Amylase	Fructosamine	Lipase	CrClearance	
GGT	Prealbumin	Albumin		
Ig G,A,M	Testosterone	Protein, Total		
C3, C4	TT4	FT4		
Folate & RBC	Cortisol	TSH		
DHEAS		Alb(U)		
SHBG		HbA1c		
		Creatinine (U)		

Table 2.3 Laboratory tests index of individuality rating [Fraser_b, 2000]

To illustrate the stratification concept, consider a test for urine Creatinine (mmol/L). A breakdown of the whole group (combined women & men), women individually and men individually for this test is detailed in Table 2.4. It shows that the specific test under discussion has an II of 0.46 when the test is applied to the whole group. However, if the whole group is stratified into separate gender groups of Men and Women, the II of the test changes dramatically. The test when performed on Women is 1.42, and when performed on Men is 1.83. Both of these II values are above the 1.4 threshold, which identifies a test which is suitable for screening or diagnosis or inter-individual monitoring. But an issue to note is the validation of the results using this technique now requires the gender of the patient to be notified to the laboratory [Collier, 2001].

Group	**CV_I**	**CV_G**	***II (Index of Individually)***
Whole Group	13.0	28.2	0.46
Women	15.7	11.0	1.42
Men	11.0	6.0	1.83

Table 2.4 CV_I, CV_G biological variation and II for urine Creatinine [Fraser_b, 2000]

Therefore, for diagnosis, screening and monitoring the ideal test will have a small CV_I and a high *index of individuality* [Collier, 2001; Fraser_b, 2000]. However, as illustrated in Table 2.4 most laboratory tests have a low *index of individuality*. This minimises the usefulness of the reference ranges for general screening and testing, unless stratification grouping is possible, and the required additional information is available to the laboratory at the time of providing the validation service.

2.4.3 Other factors complicating laboratory validation

The process of examining the validity and plausibility of clinical laboratory results is further complicated by a number of different operational issues. This subsection discusses three distinct complications that cause validation difficulty and which affect the design of the prototype ASK-agent system. They are: effect of drugs on laboratory tests, units of measurement SI units, and data storage.

Effect of drugs on laboratory tests

There are many drugs, both on and off prescription, that affect the outcome from a number of standard laboratory tests, making the generated result ambiguous. Consider for example the following five drugs [Young, 2000]:

1. Acetaminophen (Paracetamol)
2. Cephalexin (antibiotic used to treat urinary tract infections)
3. Diltiazem (used in the treatment of hypertension and angina)
4. Oral contraceptives
5. Penicillin

Each of these medications has an effect on a particular range of laboratory tests, which would alter the true result. Table 2.5 highlights the effects the above five specific medications have on common laboratory tests.

Drug	**Laboratory Test**	**Inc/Dec**	**Mechanism of Action**
Acetaminophen	Alkaline phosphates	Increase	Hepatic necrosis associated with high drug concentration
	Bilirubin	Increase	Hepatic damage with overdose
	Glucose	Decrease	GOD-PERID method, Measured by Medisense Precision QID
Cephalexin	Alkaline phosphates	Increase	Transient hepatitis and cholestatic jaundice reported
	Creatinine	Increase	Jaffe methods, Technicon SMAC, Serno Centrifichem
Diltiazem	Alkaline phosphates	Increase	Mild increases in enzyme activity
	Bilirubin	Increase	Mild transient increase
	Uric acid	Increase	May cause gout as a side effect
Oral contraceptives	Glucose	Increase	Alters glucose tolerance test
	Sodium	Increase	By salt/water retention
	Total protein	Decrease	Decreased protein synthesis; increases 3 years after administration
Penicillin	Albumin	Decrease	BCG method
	Urine Glucose	Increase	Ames Clinitest tablets
	Potassium	Decrease	If intravenous sodium penicillin infused

Table 2.5 Drug to laboratory test interference/interaction [Young, 2000]

As a consequence of consuming these medications, the laboratory results are either increased or decreased when compared to the expected result. For example, if a patient was taking Acetaminophen (*Paracetamol*) prior to a test for Alkaline Phosphates, the patient's result would be increased when compared to the same patient not taking Acetaminophen. Therefore, it is important that the laboratory is made aware of the drugs and foods consumed by the patient in order to accurately validate their results [Young, 2000].

SI Units of measurement

Standards are agreements containing technical specifications or other precise criteria to be used as rules, guidelines, or definitions of characteristics, to ensure that materials, products, processes and services are fit for their purpose. A measurement is a comparison of an unknown entity to a standard. If a single unknown object was measured by comparing it to two or more different standards, the reported unit measurement (comparison) would be different. This does not establish that the object is different, it only means that they are not compared to the same standard. Although Clinical Laboratory Standards Institute (CLSI) documents generally use units that are fully acceptable within the Système International d'Unités (SI), these do not always coincide with the units recommended by the International Union of Pure and Applied Chemistry (IUPAC) and by the International Federation of Clinical Chemistry (IFCC) for reporting results of clinical laboratory measurements. Consider for example, a serum (blood) sample used to determine the function of the liver. A reduced liver function test performed by a clinical laboratory is designed to determine the ratios of four elements in the blood sample (i.e. Bilirubin, GGT, Alkaline phosphates and Protein) [Jaeger et al., 2004; Cabot, 2004]. Table 2.6 lists these four elements and the accepted units in which they can be reported. It shows that Bilirubin results can take two forms (a) µmol/L, or (b) mg/dL, both of which are acceptable forms of reporting under the SI standard. Problems can arise when a sample is divided and issued to more than one laboratory for analysis. If one laboratory returned a Bilirubin result of 6mg/dL and the other 103 µmol/L, are these results different or the same?

Element	**Reported Units (a)**	**Reported Units (b)**	**Reported Units (c)**
Bilirubin	µmol/L	mg/dL	
GGT	µKat/L	U/L at 37° C	
Alkaline phosphates	U/L	µKat/L	U/L at 37° C
Protein, total	g/L	g/dL	

Table 2.6 Result Reporting differences [TMAP, 2006]

Using the conversion factor shown in expression (2.5) a µmol/L result can be converted into mg/dL:

$$\text{mg/dL} = (\mu\text{mol/L}) \times 0.0584 \qquad (2.5)$$

The µmol/L value can be calculated using equation (2.5) (i.e. 6mg/dL = 103 µmol/L), and so in this example they are the same, but the confusion is a real concern for clinicians.

Data storage

From a clinical laboratory standpoint it is important to store results so they can be quickly accessed and managed effectively as part of the laboratory workflow. To provide this facility, the laboratory stores result information in the form of a database entry, usually as part of the LIS. The purpose of the database is to compartmentalise data for storage and processing purposes, where the '*value'* element (e.g. 103) is stored in a different location to the '*unit*' element (e.g. µmol/L). This enables the LIS's database management system (DBMS) to add, subtract, multiply and divide the value component with relative ease, because it is separated from text. However, this separation of *value* and *units* can also result in calculation errors being generated, where *values* from different types of *units* are combined. Additional instances of data storage difficulties include: date/time stamping represented in different formats (e.g., dd/mm/yyyy, yyyy/mm/dd and mm/dd/yyyy), numeric verses narrative (e.g. numeric '10' rather than 'ten'), and consistency of medication identification (e.g. 'A*cetaminophen*' also known as '*Paracetamol').*

2.5 Guidelines and protocols

Guidelines are defined as documents published by various compliance agencies for the purpose of clarifying provisions of regulation, confirming best practice activities according to accepted procedures, and indicating how information should be interpreted [Rodney et al., 1998]. They are never considered mandatory as their purpose is to help clinicians and laboratory technologists to make decisions about appropriate actions for specific circumstances. A clinical protocol on the other hand has a more specific meaning, and is used to describe a mandatory procedure or activity that must be completed. From a medical perspective, guidelines are self-contained, standalone bodies of knowledge that focus on a medical condition, organ or disease (not on the individual patient), and can be used by medical professionals. The documents do not have any explicitly declared references, links or ties to any particular clinical laboratory *department*[7], only a disease, condition or organ, and so are technically department independent. Guidelines are deliberately structured to '*guide*' the reader on best practices,

[7] Traditional department structures within a laboratory are Microbiology/Immunology, Haematology and Biochemistry.

and improve the quality and consistency of patient care through documented evidence. Traditionally this guidance is given in the form of aims, goals, procedures, logic and facts. Through the use of guidelines, workforce actions are more consistent, predictable, and presumably of higher quality [Ellrodt et al., 1995]. For example, consider an extract for the "*criteria for testing for diabetes in asymptomatic adult individuals*" [ADA, 2006] which states that *testing for diabetes should be considered in all individuals at age 45 years and above, particularly in those with a body mass index (BMI) >25 kg/m^2 and, if normal, should be repeated at 3-year intervals*. This extract identifies a snapshot of the guideline's aims, goals, procedures, logic and fact components. The goal is that all individuals aged > 45 years should be tested for diabetes, particularly those individuals with a BMI >*25 kg/m^2*. The procedure is that the selected individuals should be tested for diabetes. The logic is Age>45 years, BMI >25 kg/m2 and Repeat Test>3 years. The facts are age, body mass index (BMI), reference range and testing intervals.

Clinical guidelines are primarily targeted at the *clinician* and not directed at the *laboratory technologists,* but to validate results and ensure that they are plausible, the *laboratory technologists* must be aware of the knowledge, aims, goals and procedures used by the *clinician.* This is because the *technologists* need to identify useful techniques, select appropriate stratification, and apply population/personal normal reference ranges in order to validate results [Collier, 2001; Fraser_b, 2000]. This sharing of information contained in guidelines and protocols provides a synergy between the two professional groups. Therefore, the best guidelines and protocols are those that expand their domain to incorporate information useful to both the *clinician* and *laboratory technologists*. Consider for example, the "*Recommendations regarding public screening for measuring blood cholesterol"* by The National Heart, Lung, and Blood Institute [NHLBI, 2004] which states "*the performance of a cholesterol test has an allowable bias*[8] *of 3.0% and an allowable coefficient of variation (CV) up to 3.0%*". This information is targeted at the laboratory technologists and sets the scene for the minimum test accuracy. If the laboratory testing procedure does not conform to this minimum standard then the rest of the guideline will be invalid. Information similar to that in the above example is delivered

[8] The ISO defines *bias* as "*the difference between the expectation of measurement results and the true value of the measured quantity*" [ISO 3534-1, 1993].

in fragmented sections throughout the guideline document, and is used by the laboratory to set population based reference ranges for the test. Another extract from the same guideline is given in Table 2.7, where it details a drug trial based on two independent coronary drugs identified as Clofibrate and Nicotinic acid. Clofibrate and Nicotinic acid are used by clinicians to aid in healthcare management in order to reduce a patient's total cholesterol level. The table extract shows that for patients taking Clofibrate the clinician would expect the Cholesterol to be 234 mg/dL when compared to a patient not on the treatment having 250 mg/dL. This would reduce the patient from a high risk to a borderline high risk.

Trial/Drug/Duration of Intervention	**Number of Subjects**	**Group**	**Total Cholesterol expressed as (mg/dL)**	**Total Cholesterol expressed as (mmol/L)**
Coronary Drug Project/ Clofibrate/5 yrs	1,103 on drug 2,789 on placebo	On-Treatment Baseline	250 234	6.5 6.08
Coronary Drug Project/ Nicotinic acid/ 5 yrs	1,119 on drug 2,789 on placebo	On-Treatment Baseline	250 226	6.5 5.87
Interpretation of results	**Note:** Baseline relates to patients on the placebo and On-Treatment relates to patients on the specific drug. All subjects are male.			
Level mg/dL	**Level mmol/L**	**Interpretation**		
<200	<5.2	Desirable level corresponding to lower risk for heart disease		
200-239	5.2-6.2	Borderline high risk		
>240	>6.2	High risk		
1	0.026	Conversion factor		

Table 2.7 Extract of Table II.3–4 from the Recommendations regarding public screening for measuring blood cholesterol [NHLBI, 2004].

The information contained in Table 2.7 relates to both the clinician and the clinical laboratory, but the guideline does not cover side-effects that are expected in patients who use this medication. These side-effects relate to the lack of effectiveness of diabetes and anticoagulant medication (Warfarin) with patients using Clofibrate, and patient urine results would be expected to contain traces of blood. These facts would be contained in other guideline literature that the clinician would be aware of and routinely access, but not the laboratory technologists. Therefore, a clinician monitoring a patient being prescribed Clofibrate would expect to have a '*blood in urine analysis*' return positive, or a poor response to diabetes specific testing (i.e. a glucose tolerance test), which is plausible for this patient as they are taking the medication. However, the positive result of the '*blood in urine'* or poor responses to diabetes medication would be a concern for the laboratory (in the absence of the aforementioned information), and thus the result would almost certainly be flagged for further scrutiny. This is another example where there is a lack of information cross-pollination between expert groups caring for patients, because a formalised structure wasn't in place to log this information and is not part of an automated validation system.

Laboratory guidelines and protocols on the other hand are documents that are relevant particularly to the *laboratory technologists* and not to the *clinician* [Oosterhuis 2004]. They are directed at quality control, testing procedures, staff training, result generation and services provided by the clinical laboratory for managing the delivery of patient results. These guideline documents also reaffirm information disseminated through clinical guidelines, but directed at *technologists* [Oosterhuis 2004] for inclusion within their laboratory's Standard Operating Procedures (SOP) and Quality Action Plans (QAP). The content provided by these guidelines is presented in a similar fashion to the clinical guidelines where they provide details such as aims, goals, procedures and normal reference ranges. So in summary, guideline information is not as useful as it could be until it is accessible in its integrated form to all decision makers using the service. Developing a technique that permits both clinical and laboratory guidelines and protocols to be encoded and used, is of critical importance if islands of information are not to be produced.

2.6 Laboratory information systems

Laboratory Information Systems (LIS) is a software application, or more specifically a group of software applications, which are used by laboratory staff to manage and coordinate their day to day activities [Burtis et al., 1998]. Typical functions performed by LIS's are [CAP, 2005; Harmening, 2003; Van Bemmel et al., 2002]:

1. Store patients' past and present laboratory results.
2. Store sample specific information, such as the date/time stamps of sample taking, entering the laboratory and type of sample.
3. Store patient specific information (i.e. ID, weight, height and gender etc.) for use during the interpretation of the laboratory results.

4. Store quality control data, such as instrument calibration, laboratory throughput, and control sample results etc.
5. Generate reports for presenting patient results.
6. Generate reports for laboratory management on throughput, quality control and error reporting.
7. Provide a facility to interpret laboratory test results.
8. Provide the management of laboratory tests and reporting workflows.

The teams of professionals managing large clinical laboratories develop the functions of the LIS to support and complement their laboratories' core activities (i.e. the measurement, validation and reporting of patient sample results). In practice the LIS is realised by separating the key functions into dedicated software applications [CAP, 2005]. In most cases the storage aspects of the LIS are managed by database systems, and interpretation of the laboratory results are completed using database management rules, or commercially available specialist laboratory management systems. These specialist systems observe the information stored in the laboratory database slots and use various types of logic and knowledge to validate the results [Van Bemmel et al., 2002]. Therefore, in essence a modern LIS comprises the following processes:

1. ***A method of describing and representing the context of data, so it can be accurately and correctly interpreted.***

The LIS must have provision to store atomic data (e.g. ID, weight, height, and gender), and also permit higher levels of data abstraction to be described and represented (e.g. glucose tolerance, body mass index, or classification of diabetes).

2. ***A method of communicating the data in a manner, which preserves its context as described in (1) between modules.***

When two or more modules communicate data it is vital that all systems have identical understanding of the message content.

3. ***A method of representing and executing logic and/or workflow, which can be executed by the system's interpretation/inference engine.***
   The LIS must have the ability to reproduce electronically the knowledge, logic, and workflows contained within laboratory guidelines and protocols and manage their execution.

These three generic processes are repeated in various forms throughout the LIS as described in narrative form by Harmening and Van Bemmel, and interpreted graphically in Figure 2.5 [Harmening M, 2003; Van Bemmel et al., 2002]. The data context components are illustrated using the colour gold and act as an information interpretation service between the client, analyser and database communications and the control system. The communication components are illustrated using the colour red and include the client, analyser, database communication and inter-module communication. Finally the logic components are illustrated using the colour salmon and are the control system, inference engine, SOP work list and workload planner etc.

Although guideline-based decision support systems and the ASK-agent primarily focus on the interpretation/inference engine resource, the method of describing the context of data and its communication between separate modules must also be examined. This is because the interpretation/inference engine resource is predominately dependent on the other two facilities.

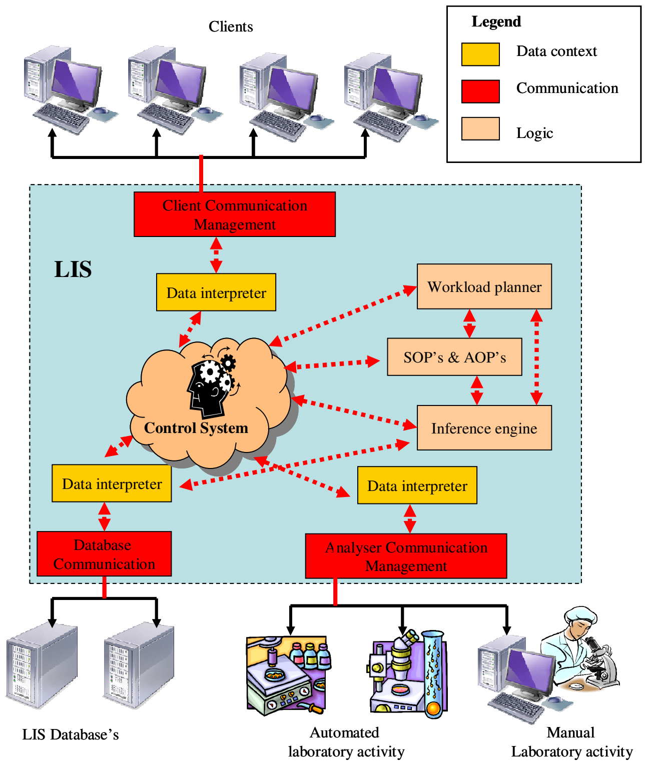


Figure 2.5 Laboratory information system components

The traditional method of developing guideline-based decision support is to identify key workflow paths (i.e. SOP and QAP) and trends, which complement the core activities of the clinical laboratory. The laboratory then develops these workflow paths into customised computerised procedures. These workflow paths incorporate aspects of institution policies, guidelines and protocols to form a laboratory procedure [Bachmann, 2000]. This laboratory procedure is a localised standard to which all laboratory testing of a specific type must comply. Each laboratory site has unique combinations of technical staff, equipment, sample types and reporting requirements, and therefore the implementation of the guideline is site specific. Site specific guidelines cannot be easily implemented by other medical institutions, have poor reusability, usually lack cross-compliance and have limited scalability. Cross-compliance relates to the ability of a laboratory to validate a patient sample in a manner that is satisfactory to a number of different certification bodies. The remainder of this section is divided under two headings: guideline based decision support, and generic text guideline to Computer Interpretable Guideline (CIG) conversion.

Guideline based decision support systems

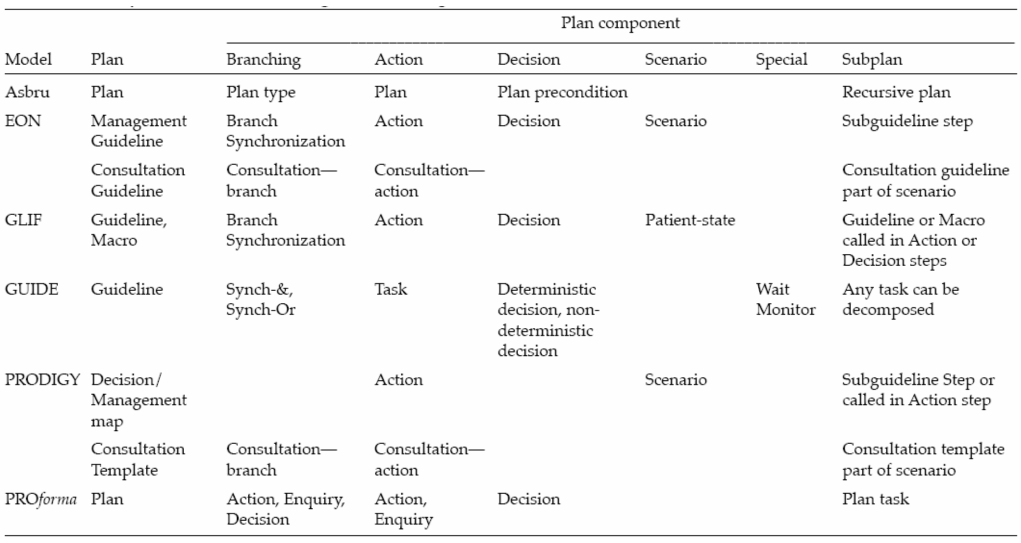
A prerequisite for developing guideline-based decision support systems is the ability to create computer interpretable representations of the knowledge and logic contained within clinical guidelines [Peleg_d et al., 2003]. One organisation achieving this is the "*National Guideline Clearinghouse*™" (NGC, http://www.guideline.gov/), which is a public resource for evidence-based clinical practice guidelines. The NGC was an initiative by the *Agency for Healthcare Research and Quality* and the *U.S. Department of Health and Human Services,* and developed an Extendable Mark-up Language (XML) standard for the layout of clinical guidelines. XML allows developers to set standards by defining the structure and sequence of information that should appear in a document, but not the technical content. XML, in combination with other standards, makes it possible to define the content of a document separately from its formatting, making it easy to reuse that content in other applications, or for other presentation environments. Most importantly XML provides a basic syntax, which can be used to share information between different kinds of computers, applications, and organisations. The NGC standard format is designed for guideline content presentation using various XML filters, and is available in a form that can permit the XML documents to link together, use logic or collaborate if these facilities were developed by others.

A number of other groups are actively developing more sophisticated CIG models, which support an increased level of electronic guideline representation [Peleg_d et al., 2003]. These include CIG models such as Asbru, EON, GLIF, GUIDE, PRODIGY, and PROforma. A brief synopsis of each is given in Table 2.8.

Item	Title	Description
1	Asbru	Asbru is a time-oriented, intention-based, skeletal-plan specification language that is used to represent clinical protocols. Skeletal plans capture the essence of a procedure, but leave scope for execution-time flexibility in the achievement of particular intentions by combining plans [Shahar et al., 1998].
2	EON	EON provides a suite of models and software components for creating guideline-based applications. It views the guideline model as the core of an extensible set of models, such as a model for performing temporal abstractions. EON uses a task-based approach to define decision-support services that can be implemented using alternative techniques [Tu et al., 2001].
3	Guideline Interchange Format version 3 (GLIF)	The GLIF stresses the importance of sharing guidelines among different institutions and software systems. GLIF builds on the most useful features of other guideline models, and incorporates standards used in healthcare. Its expression language was originally based on the Arden Syntax, and is the default medical data model based on the HL7 [Peleg_c et al., 2004; Peleg_d et al., 2003]. This model uses the object-orientated approach.
4	GUIDE	GUIDE supports integrating modelled guidelines into organisation workflows, using decision analytical models such as decision trees and influence diagrams, and simulating guideline implementation in terms of Petri nets. GUIDE considers issues such as patient data, the implementing facility's organisational structure, and resource allocation [Ciccarese et al., 2004].
5	PRODIGY	PRODIGY provides support for chronic disease management in primary care. The PRODIGY project's aims are to produce the simplest, most readily comprehensible model necessary to represent this class of guidelines. Teams of clinicians have used Protégé's knowledge engineering environment to encode three complex chronic disease-management guidelines [Johnson et al., 2001].
6	PROforma	PROforma combines logic programming and object-oriented modelling. PROforma supports four tasks: actions, compound plans, decisions, and enquiries. All tasks share attributes describing goals, control flow, pre-conditions, and post-conditions [Sutton et al., 2003].

Table 2.8 CIG models.

When the underlying structures of these six CIG models are examined more closely, the use of three main knowledge and logic encoding methodologies emerge. These are rule-based, guideline mark-up languages, and object-oriented guideline languages, which are described in more detail in chapter 3. It was also found that none of these CIG models incorporate any communication directly between guidelines in their model architecture either explicitly or implicitly, other than direct linking or nesting, where the strand of control is handed over to another guideline [Peleg_d et al., 2003]. In addition to the knowledge and logic encoding similarities, a set of standardised guideline conversion features also begin to emerge. In Table 2.9 a number of these features specifically relating to guideline *plans* and *action* components are categorised. This table shows the main *'guideline plan'* operations used by CIG's are Scenario/Patient State plan, Branch Steps plans, Synchronisation _Steps plans, Wait_Step plans, Monitor task plans and Entry Points into Guideline Plans.



		Plan component					
Model	Plan	Branching	Action	Decision	Scenario	Special	Subplan
Asbru	Plan	Plan type	Plan	Plan precondition			Recursive plan
EON	Management Guideline	Branch Synchronization	Action	Decision	Scenario		Subguideline step
	Consultation Guideline	Consultation—branch	Consultation—action				Consultation guideline part of scenario
GLIF	Guideline, Macro	Branch Synchronization	Action	Decision	Patient-state		Guideline or Macro called in Action or Decision steps
GUIDE	Guideline	Synch-&, Synch-Or	Task	Deterministic decision, non-deterministic decision		Wait Monitor	Any task can be decomposed
PRODIGY	Decision/ Management map		Action		Scenario		Subguideline Step or called in Action step
	Consultation Template	Consultation—branch	Consultation—action				Consultation template part of scenario
PRO*forma*	Plan	Action, Enquiry, Decision	Action, Enquiry	Decision			Plan task

Table 2.9 Terms used by CIG models for Plans and Actions [Peleg_d et al., 2003]

Table 2.9 Terms used by CIG models for Plans and Actions explanation:

The *Plan component* refers to functions that are used to reproduce key characteristics of a guideline, such as branching down parallel paths, or perform an action (i.e. blood test, give medication). The six CIG models use different terminology to refer to the various types of plan components. However, with minor variations in implementation, all formats represent decisions, actions, and nested sub-plans. This table also includes various other features used by CIG's, such as scenario, patient state and wait monitor.

Generic text guideline to CIG conversion

Although each group has adopted different conversion methodologies, reflecting their members' interests and expertise, many of the approaches share a hierarchical decomposition of guideline information and logic into nested primitive tasks. Each primitive task is considered atomic where it represents a single rule or activity of the guideline. The guideline knowledge is represented by linking the primitive tasks into a larger task set, and is accessed by branching down through the nested primitive tasks over time. This approach has been described as the “*task-based paradigm*” or Task-Network Models (TNM) [Peleg_d et al., 2003; Fox et al., 2000]. The Arden Syntax is perhaps the best-known language for representing clinical knowledge in patient specific decision-support systems. It is a rule-based formalism developed for encoding individual clinical rules as Medical Logic Modules (MLMs).

Each Arden Syntax MLM is written to behave like a single rule or activity, as instructions within an MLM are executed in an ordered series until an outcome is reached. To realise this each MLM uses four main "slots": an "*evoke slot*", a "*data slot*", a "*logic slot*" and an "*action slot*" [Peleg_b et al., 2001]. The *evoke slot* acts as the "trigger" (e.g. upon admission of the patient), and the *data slot* as a retriever of the relevant data from the LIS (i.e. 'has_Clinical_CHD', or 'Date_Last_Cholesterol_Test'). The *logic slot* uses this data in an if-then-else type arrangement, where the "IF" part of the logic statement is the logic test (e.g. IF('Current_Date' -'Date_Last_Cholesterol_Test') >= 5 years), and then the *action slot* executes the "then" piece of the statement. A triggering event can be simply a diagnosis LIS entry, or another data variable, such as a laboratory test result, written to the LIS. Other MLM triggers used can be the passage of time, a call from another MLM (such as a link), or abnormal data. This approach is termed *direct coupling* where a guideline task acting on behalf of a piece of specialised knowledge can switch the control thread to another guideline task [Kavanagh et al., 2002]. All of these can produce the *action slot* to deliver a reminder, or an alert based on the data (e.g., to administer a follow-up preventative exam), or a specific guideline recommendation (e.g., dosage of medication).

In addition to MLM’s, Augmented Decision Tables (ADTs) have also been adapted to represent clinical knowledge in patient specific decision-support systems [Peleg_d et al., 2003]. ADTs go beyond Arden’s rule-based and *direct coupling* formalism by augmenting decision-table rules with additional information, such as probability and utility information, in a similar fashion to quantitative scoring. Although the Arden Syntax has been adapted in the majority of guideline representations by employing interacting MLMs, neither MLMs nor ADTs provide full support for conceptualising a multi-step guideline that unfolds over time. The TNM approach has arisen in response to this problem. The languages used by the TNM approach provide modelling primitives specifically designed for representing complex, multi-step clinical guidelines and for describing temporal and other relationships between component tasks. Unlike rule-based systems, TNMs can explicitly model alternative pathways or sequences of tasks (i.e., control flow), and they provide tools for visual representation of plans and the organisation of tasks within them. A more commonly known incarnation of the TNM approach from a computer science standpoint is the blackboard system (see section 3.7.2).

If the ASK-agent model is to be considered an alternative to the CIG model, it must be able to implement each CIG component of functionality, in addition to allowing true problem solving collaboration, using a more open communication technique. The dimensions and sub-dimensions presented in Peleg's paper [Peleg_d et al., 2003] will therefore form the basis of a comparison between the existing suite of CIG models and the novel ASK-agent approach later in this thesis.

2.7 Choice of medical domain

Medical institutions have an ethical and legal responsibility to provide correct and appropriate care for their patients, but unfortunately taking a real-world perspective this care is usually delivered within substantial resource and budget constraints. In general, specialities in medical practice permit the medical institute to direct healthcare around a patient's most significant illness (e.g. Neurology, Immunology, Haematology, Gastroenterology, Cardiology and Dermatology). Not all hospitals provide the full range of these specialities; therefore patients can be moved to another institution that better suits their requirements after they have been assessed [Plebani & Carraro, 1997]. The function of a clinical laboratory is to act in a supporting capacity to the main core activities provided by the medical institution and the medical professionals therein. The laboratory department structure therefore, depends on hospital core activities, and political/financial pressures. This means that the range of analysis and services each department provides can change dramatically as they evolve over time. In some cases test duplication between departments occurs [Rao et al., 2003]. The traditional department structures within a laboratory are Microbiology/Immunology, Haematology and Biochemistry, which are described in Table 2.10, but other departments do exist.

Department	**Description**
Microbiology /Immunology	Microbiology is the study of micro-organisms, which are unicellular, or cell-cluster microscopic organisms. This includes eukaryotes such as fungi and protists, and prokaryotes such as bacteria. Viruses, which are not strictly classed as living organisms, are also studied in this department.
Haematology	Haematology is the study of blood, the blood-forming organs, and blood diseases. Haematology includes the study of etiology, diagnosis, treatment, prognosis, and prevention of blood diseases.
Biochemistry	Biochemistry is the study of chemical processes and transformations in living organisms. It focuses on the structure and function of cellular components, such as proteins, carbohydrates, lipids, nucleic acids, and other bio-molecules. Chemical biology uses tools and technology developed within synthetic chemistry to gain an insight into world of biochemistry.

Table 2.10 Clinical laboratory departments

In order to ensure that the ASK-agent system can accurately represent clinical and laboratory guidelines and protocols it is important to choose a domain that can fully test a validation prototype. Therefore, the choice of medical domain in which to demonstrate the ASK-agent approach must be carefully selected to emphasise the generic, yet difficult, aspects of clinical laboratory result validation. The main criteria for selecting an appropriate medical domain are:

1. A reliable and consistent set of standard analyte results should be utilised in order to act as a point of reference for the ASK-agents validation technique. These results should be obtained from a well established group of tests or procedures with standardised definitions for specific sets of results used in a clinical laboratory setting. It is important that the results are not based on new testing regimes, because the meaning and significance of the reported results will not have been fully established and could prejudice the testing of the ASK-agent's approach.

2. Some laboratory analysis techniques are based on an interpretation of the sample by the operator (e.g. microscope inspection of a biopsy). The reported results largely depend on the biopsy sample being a reasonable representative

sample of the examined substance (as not all biopsies are homogeneous), and the experience of the operator [Levinson et al., 2002]. Control samples cannot be used and these types of analysis techniques tend to produce wider variations in reported results, which make independent result validation difficult to achieve [Fraser_b, 2000]. Other laboratory testing procedures employ commercial analysers (e.g. for serum blood or urine) that make use of control samples which produce a narrower variation in the reported results and can be validated using numerical examination, such as control sample comparisons, delta checking, gender/age reference ranges etc. Therefore, to ensure consistency in validation for demonstration purposes the domain must use procedures that can be reproduced using different types of numerical techniques.

3. The validation procedure should support multi-validation paths which can be used to dynamically choose an appropriate route to validate. This would permit the ASK-agent have a form of autonomous control over its activity. Each path should utilise different types of information (e.g. gender and age, or previous two results less than a month old, etc.), or techniques (e.g. statistical, rule-based etc.) which can be used to validate the same result but by using different modus operandi. Although fixed mandatory actions can be implemented they reduce the ability of the ASK-agent to act autonomously and are better handled using other software techniques, such as procedural or rule based approaches.

The search for an appropriate medical domain focuses on three main organs of the body, namely Heart, Kidneys and Liver. The medical domain focusing on the Heart, specifically in relation to laboratory result validation is limited, because the principle methods of assessing the Heart are ultrasound, x-ray, resonance scanning etc. Outcomes using these methods are not always based on numerical analysis, and therefore difficult to implement using CIG and ASK-agent. In addition to the lack of numerical analysis, the activity of the Heart is to circulate blood around the body. This activity does not truly interface with different components of the human body such as blood, urine, digestion, etc. Results used in the Kidney domain are currently available in the laboratory without any alteration to work practices, and are mostly validated using numerical analysis [Casiday et al, 2001]. The domain is based on different interfaces with components of the body specifically blood, urine and bowel movement. Therefore the medical domain focusing on the Kidneys, specifically in relation to laboratory result validation makes it a very appropriate for an agent validation system. But an even stronger medical domain based on the selection criterion is the Liver domain. The laboratory currently generates results used to evaluate the Liver based primarily on pure numerical analysis [Cabot, 2004], although some ultrasound analysis can take place. The Liver interfaces with the blood, urine and digestive system, and its validation techniques can be separated into autonomous systems [Jaeger et al., 2004; Cabot, 2004]. The domain is well documented and accepted definitions for specific sets of results are available. For the purpose of demonstrating this novel ASK-agent approach the Liver domain emphasises a substantial number of complexities involved in performing laboratory result validation from multiple domains, and is therefore the most appropriate medical domain to use.

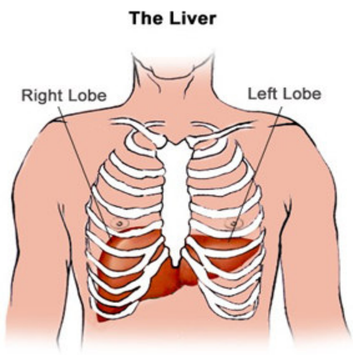


Figure 2.6 Human Liver [Medicinenet, 2006]

The liver is the largest solid organ in the body, located just below the diaphragm as shown in Figure 2.6, and plays an important role in the human metabolism by performing the following functions [Jaeger et al., 2004; Cabot, 2004; Medi06_a]:

1. Manufacture (synthesise) proteins, including albumin (to help maintain the volume of blood) and blood clotting factors.

2. Synthesise, store, and process (metabolise) fats, including fatty acids (used for energy) and cholesterol.
3. Metabolise and store carbohydrates, which are used as the source for the sugar (glucose) in blood used by red blood cells and the brain.
4. Form and secrete bile that contains bile acids to aid in the intestinal absorption of fats and the fat-soluble vitamins A, D, E, and K.
5. Eliminate, by metabolising and/or secreting, the potentially harmful biochemical products produced by the body, such as Bilirubin from the breakdown of old red blood cells and Ammonia from the breakdown of proteins.
6. Detoxify, by metabolising and/or secreting, drugs, alcohol, and environmental toxins.

Due to the multiple functions performed by the Liver, it is possible for a patient to present to a clinician with a number of medical complications, which are traditionally associated with the Liver. But it is difficult to identify if the Liver is the source of the illness or impaired as a result of illness elsewhere in the body. When the *Liver Function Test (LFT)* is completed the results give the *clinician* an indication if Liver malfunction is the source of the complications. The name *Liver function test* is a misleading title for the test, as this does not actually measure the '*function*' of the Liver. The test is more an indicator for the integrity of the Liver cell membranes. Blood samples are analysed for levels of specific enzymes in the blood stream, of which there are generally 5–7 specific analytes that are measured (depending on the institute and doctor involved). Each of these analyte tests have an index of individuality of <0.99, but by combining them together their index of individuality increases greatly, which is achieved by optimising the signal-to-noise ratio of the tests [Petersen, 2005]. The signal-to-noise ratio refers to the ratio of useful information to false or irrelevant data. Collectively these tests are called a *Liver Function Test* (LFT). These tests are listed in Table 2.11, with a detailed explanation given in Appendix D. The measured enzymes are referred to as 'markers' of disease and dysfunction. Cholesterol which is mostly synthesised in the Liver is also used by physicians to provide additional information on the functionality of the Liver, but is not strictly included in the LFT. For completeness serum Cholesterol measurement is included here, and also in Appendix D, because of its direct relevance to the Liver's operation, and is used for illustration purposes in further chapters. The proof of concept model focuses on adult only ranges and classifications.

Item	Analytes	Test Index of Individuality	Description
1	Alkaline Phosphates	<0.4	Produced in the cells of the Liver and bone with some activity in the intestine, placenta during pregnancy, and kidneys. Used as a marker for tumours, but also present in elevated levels in bone injury, pregnancy, or skeletal growth.
2	GGT (Gamma-Glutamyl Transpeptidase)	<0.4	GGT is involved in the transport of amino acids and peptides into cells. GGT is mainly found in Liver cells and is extremely sensitive to alcohol use.
3	Bilirubin	0.6-0.99	Bilirubin is a by-product of the breakdown of red blood cells in the Liver. Bilirubin is a good indication of the Liver's function. Excreted into the bile, Bilirubin gives the bile its pigmentation.
4	ALT (Alkaline aminotransferase)	0.4-0.59	ALT is an enzyme produced by Liver cells with lesser amounts in the kidneys, heart, and skeletal muscles, and is a specific indicator of acute Liver cell damage.
5	Total Protein	0.6-0.99	Proteins are the most abundant compound in serum. The major serum proteins measured are albumin and globulin.
6	Albumin	0.6-0.99	Albumin is the major constituent of serum protein (usually over 50%). It is manufactured by the Liver from the amino acids taken through the diet.
7	Globulin	0.6-0.99	Globulin, a larger protein than albumin, is important for its immunologic responses, especially its gamma portion. Globulins have many diverse functions such as, being the carrier of some hormones, lipids, metals, and antibodies.
8	Cholesterol	0.4-0.59	Serum Cholesterol is a critical fat that is a structural component of cell membrane and plasma lipoproteins, and is mostly synthesised in the Liver. Two types of cholesterol are measured by analysers: high density lipoproteins (HDL), and low density lipoproteins (LDL) for which Triglyceride measurement is used. LDL is the cholesterol rich remnants of the lipid transport vehicle VLDL (very-low density lipoproteins) and HDL is the cholesterol carried by the alpha lipoproteins.

Table 2.11 Liver Function Test (LFT)

Each analyte is described in more detail in Appendix D, where the information and data was interpreted using Carey, Harrison, Jaeger, Cabot and Johnston, and their contribution is duly noted [Carey, 2006; Harrison, 2004; Jaeger et al., 2004; Cabot, 2004; Johnston, 1999].

2.8 Discussion

This chapter has illustrated that clinical laboratory results play a pivotal role in aiding clinicians to make decisions about patients in their care. Therefore the most important activity the laboratory performs is to ensure that result 'snapshots' are valid and plausible for the specific patient, before clinicians can rely on them to make healthcare decisions. Section 2.2, showed the difference between existing validation procedures, which are process centred and the patient-centred validation which is centred on the individual patient. This alteration to the dynamic of execution improves the quality of patient care by crafting a set of validation procedures around known information to maximise its accuracy for a particular patient.

Section 2.3 provided an overview of the complete laboratory validation cycle, which was expanded in section 2.4, to establish that the key influencing factor in providing patient-centred validation is achieved through the management of the biological variation factors. This was because the outcomes from the other factors of the validation service do not use, and are largely unaffected by, patient specific data. Section 2.4 also introduced the concept of laboratory test *index of individuality* (II), and showed that the majority of them have an II < 1.4, and therefore have a marked individuality. The entire group of LFT's each have an II <0.99. In the cases where the analyte tests have a *marked individuality,* such as the LFT's, stratification can be used to group sections of the population together to increase the II. This can make it increasingly difficult to validate patient results, unless more patient specific information is known at the time of providing the validation service.

Section 2.5 demonstrated that the primary conduit for the dissemination of biological variation information is through clinical and laboratory guidelines and protocols. Clinical guidelines are primarily targeted at the *clinician* and not directed at the *laboratory technologists*. They are used by the *clinician* to support and complement the application of their expertise during the interaction with a specific patient. But in order to validate results that are used as part of these guidelines, and to ensure they are plausible for the patient, the *laboratory technologists* must be aware of the knowledge, aims, goals and procedures in order to identify useful techniques, appropriate stratification criteria, and population/personal normal reference ranges. Laboratory guidelines and protocols on the other hand are documents that are relevant particularly to the *laboratory technologists* and not to the *clinician.* They are directed at quality control, testing procedures, staff training, result generation and services provided by the clinical laboratory for managing the delivery of patient results. But both clinical and laboratory guidelines present their content in similar ways, where they provide information such as aims, goals, procedures and normal reference ranges, and could therefore be managed by the same software product.

The final concluding comment relates to methods of translating narrative guidelines and protocols into computer executable symbols. Existing CIG's develop comprehensive methods of translating narrative styled guidelines and protocols into computer executable symbols. These methods have a tremendous reuse value that can be exploited to aid the advancement of the proposed agent system. To realise this potential it is important to emphasise the difference between *"adapting and modifying"* the execution of a guideline, which is an *intra*-guideline activity, and *"adapting and modifying"* how the guidelines and protocols themselves relate to one another, which is an *inter*-guideline activity. To clarify the difference between intra-guideline and inter-guideline activity, consider the example where a *clinician* follows two or more guidelines that relate to a specific patient. The *intra*-guideline activity is based on observing the rules and information contained within a guideline, which should not change; otherwise there would be a doubt if the guideline is being followed. The *inter*-guideline activity on the other hand relates to the selection of the most appropriate guidelines to use, based on how the guidelines and protocols relate to one another and to the patient. This separation of *intra* and *inter*-guideline activities permits the reuse of existing CIG translators for narrative styled guidelines and protocols. These translators which act as maps from narrative guidelines into computer executable symbols can be reused, without necessarily having to also use the CIG's centralised engine for managing *inter*-guideline activity.

Chapter

3

Overview of agent software and collaboration software

3.1 Introduction

The aim of this chapter is to provide an overview of agent software and collaboration software, so that a system that can support *patient-centred* laboratory validation services can be developed. Chapter 1 introduced agent-oriented architecture as software that synthesises traits of human intelligence using elements, such as attributes (beliefs), methods (plans) and desires (goals) [Rao et al., 1995]. This thesis demonstrates that software agents are better suited to reproduce the function of clinical and laboratory guidelines and protocols than traditional object-oriented (OO) based computer interpretable guideline (CIG) architectures. This is because a synergy exists between knowledge base, plans, decisions, action, goals and the self-contained operation dynamic between guidelines and agents, which are not as consistently presented within the OO approach. Section 3.2 compares object-oriented and agent-oriented approaches as a solution to computerising clinical and laboratory guidelines.

Modern software applications provide intelligence in two ways: intelligence offered to the process (i.e. the ability to solve a specific problem) [Turban_a et al., 2005], and intelligence offered through communications (i.e. the ability to communicate specific information in a particular context and time) [Schumacher, 2001]. The agency approach encapsulates both types of intelligence. Due to the importance intelligence has in relation to the operation of agents, computer intelligence through inference logic reasoning is further developed in section 3.3, where different approaches are discussed that reproduce aspects of human reasoning.

Although the concept of the agent is based on the agency principle, there are a number of different approaches and platforms available. On examination of a selection of these agent platforms it was observed that some are at an advanced stage of development and are targeted at a large audience base, not just the medical domain. Therefore, selection of an appropriate *base-level* agent platform with an ability to be adapted or customised in order to develop the proposed ASK-agent is essential. Section 3.4 refines the theory and operation of distributed agents, and introduces five different implementations.

The agent approach is primarily a mechanism for distributed processing, where the activity of modules is distributed throughout the system [Braubach et al., 2004]. Therefore, the system largely depends on each autonomous module being able to communicate domain specific knowledge effectively and efficiently. Section 3.5 introduces agent communication in order to preserve the meaning when conversing between two or more systems.

The construction of a modern software application can involve many stakeholders and computer modelling has the potential to manage development activities among them [Benson et al., 1998]. Section 3.6 examines three architectures (i.e. Agent Unified Modelling Language (AUML), MaSE, and Prometheus Modelling Languages) designed specifically for modelling agent systems and examines their role in the design of the ASK-agent.

The inherent autonomous action of an agent can hamper its ability to collaborate on achieving a task with other agents. This is primarily due to the lack of centralised harmonising control over separate modules. In section 3.7 the concept of collaborative software is introduced and is expanded in terms of the proposed ASK-agent.

The social sciences have developed many tools which can be used to analyse human collaboration. Although these tools do not transpose directly to software they can be used to analyse collaboration in agent based social groups. Section 3.8 provides an overview of the activity theory, and develops a technique to improve the collaboration capability of the ASK-agent.

Finally section 3.9 presents various concluding remarks.

3.2 Overview of object-oriented and agent-oriented approaches

Object-oriented (OO) programming is a *software paradigm*, where the software programmer not only defines the type of a data structure, but also the types of operations (or functions) that can be applied to the data [Luger_c, 2002; Pressman_c, 1997]. The OO paradigm centres on describing both *methods* and *attributes* through components called objects [Turban_a et al., 2005]. For example, a person object can have attributes (e.g.

height, weight, age and sex), but also have methods (e.g. get food, eat food, climb stairs). The motivation to manipulate objects to achieve the desired outcome is expressed within computer program code, through the calling and appropriate use of these objects.

The agent-oriented approach operates on the notion that human intelligence and decisions can be synthesised using tangible elements, such as beliefs, desires and intentions (BDI) (i.e. beliefs, desires, goals, sub-goals, plans and intentions) [Rao et al., 1995]. Although, actual human brain execution is a parallel process using synapses firing and chemical reactions, the overall output for particular human actions can be represented using a more straightforward model. In the agency approach, human decisions are based on known facts (beliefs), background knowledge in the form of partial solutions (sub-plans) and desires (goals), which combine together to form an algorithm (i.e. have weightings and relationships) which provides artificial intelligence [Braubach et al., 2004; Rao et al., 1995]. Depending on the weightings of respective plans and goals that can be selected during run-time, some aspects of human intelligence are reproduced. The agent-oriented paradigm advances the OO paradigm by adding a formalised description for desire (or motivation) in the form of goals, rather than have the desire implicit in the OO code. For example consider the previous person object, where a motivation element (e.g. "if hungry") is introduced in the agent module itself, and can trigger the "get food" method, which subsequently triggers the "eat food" method, thus fulfilling the motivation. This departure of the motivation or *desire* element being implemented through formal/explicit goals, rather than implicit developed code offers two refinements to the execution:

1. It formally expresses motivation through goals/desires.
2. It permits an agent to query another agent's goals/desires, to ensure there is a consistency in *desire* ontology.

These self-contained autonomous agents are often deployed in environments where they interact, and perhaps cooperate, with other agents (including both people and software) that possibly have conflicting goals, and such environments are known as multi-agent systems (MAS). However, both object-oriented and agent-oriented approaches primarily operate by dividing an application up into separate modules, and as a result of choosing either of these approaches there is an increased demand on networking and communication between modules of respective systems.

3.3 Software intelligence.

Inference is defined as the process of deriving a conclusion based solely on logic and information that is already known [Luger, 2002]. The main purpose of inference logic is to simulate some trait of human decision-making or reasoning in a software application in order to provide some '*intelligence'* [Turban_a et al., 2005]. *Intelligence* is defined as a property of the human mind that represents combinations of elements such as thought, perception, emotion, will and imagination. It provides the brain with the mental ability to reason, plan, solve problems, to think conceptually, understand ideas and language and the capability to learn. Developers have sought to reproduce aspects of human intelligence artificially through software applications to enable computers complete increasingly complex tasks. Figure 3.1 depicts the software evolution timeline by Pressman [Pressman_a, 1997], but expanded to include the fifth era from 2000 to the current year. This timeline demonstrates that the development of intelligence in modern computer systems is two fold: intelligence offered by the process, and intelligence offered through communications.

Process intelligence relates to the ability to reason, plan, solve problems, to think conceptually, understand ideas and language and the capability to learn from within the software application (intra-software). The intelligence of communications relates to meaningful usage of inter-software module communiqué. Where planning, reasoning and problem solving can be accomplished at a *'network communication level'* (inter-software), and not solely within an application [Vinoski, 1997]. Examples of communication intelligence are choosing the printer with the least workload when printing a document, or choosing the nearest strongest WI-FI carrier when more than one is available.

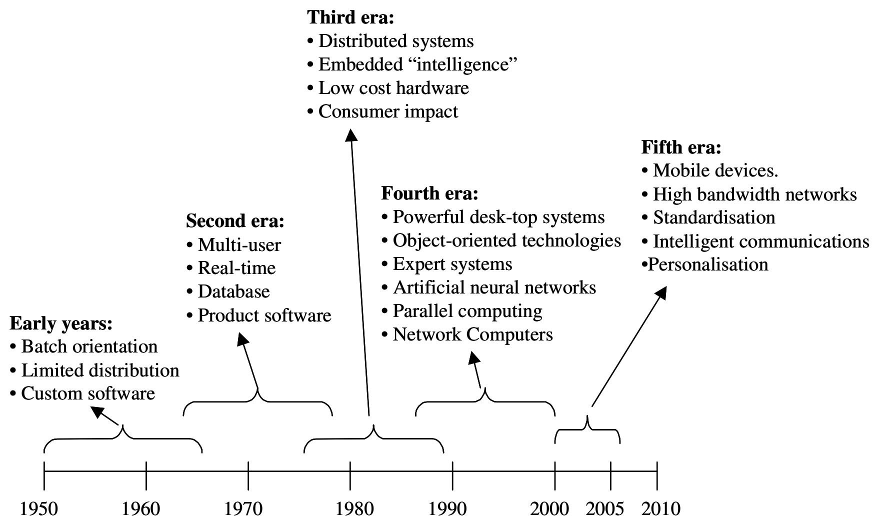


Figure 3.1 Evolution of software/hardware timeline [Pressman_a, 1997, modified]

Figure 3.1 Evolution of software/hardware timeline explanation:

The timeline presents "e*arly years*" where custom-built and batch-oriented software systems were the norm. Software was developed and executed in a primitive manner and hardware was expensive. In the "s*econd era*", evolution of software design allowed developers to reuse previously implemented software, and provide various generic products. As Intel produced the first commercial microprocessor in 1971, hardware costs began to fall and more processing power became available. Database technology advanced and exploited real-time, multi-user processing systems. In the "third era" the lowering of hardware costs through sector competition allowed for processing power to be distributed, thus distributed software became a possibility and embedded "*intelligence*" began to evolve. The "*fourth era*" focused on the utilisation of computer networks to harness distributed computational power, and object-oriented technologies and artificial intelligence systems began to be realised. The "*fifth era*" shows an increased reliance on mobile devices, (e.g. mobile phones and portable computers), and connecting these devices together using higher bandwidth and intelligent communications.

The principal additions made by the agency approach, which belongs to fifth era technology, over the distributed OO approach, which belongs to fourth era technology, can be realised by comparing the generalised distributed software application structure, which is illustrated in Figure 3.2. In this figure a clear distinction is made between the intra-software process intelligence, and the inter-software communication intelligence which are both forms of artificial intelligence. The fundamental modifications that are made by the agent applications focus on the inference-engine, message-envelope and communication facility, and all the terms are discussed briefly below.

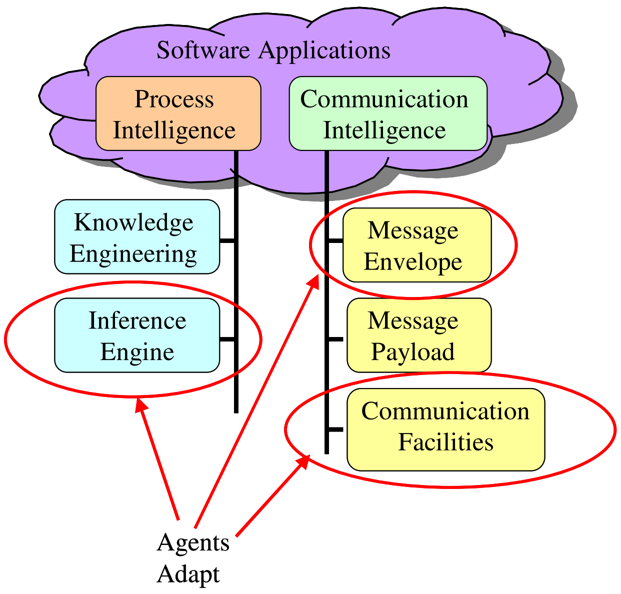


Figure 3.2 Generalised distributed software application structure

Knowledge Engineering

The term knowledge engineering relates to the structure in which knowledge is stored within the software application. Traditional forms of knowledge storage are: frames, objects, scripts, semantic networks and nodal trees. Both OO and agent-orientated applications use similar forms of knowledge representation.

Inference engine

The inference engine relates to the process of deriving a conclusion, or executing an action based solely on information, which is already known, and usually stored within the knowledge representation, or a received message or procedural call. It provides modules with the ability to reason about information in the knowledge base, for the ultimate purpose of formulating new conclusions or choosing an appropriate action. A number of *reasoning/inference* methods used in modern inference engines are summarised in Table 3.1, which extends Turban's synopsis [Turban_a et al., 2005] to include modal logic reasoning and hybrid reasoning.

Method	**Summary**
Formal reasoning	Syntactic manipulation of data structures to deduce new facts following prescribed rules of inferences.
Inductive reasoning	Move from some established facts to draw general conclusion.
Analogical reasoning	Derive solution using a known analogy. Use of similar past experience.
Deductive reasoning	Reasoning where the conclusion is necessitated by or reached from previously known facts.
Procedural (numerical) / Temporal reasoning	Use of mathematical models or simulation, and comparing current inputs against expected inputs.
Meta-level reasoning	Reasoning and deliberating about the most appropriate actions to commit to, from a selection of possible actions based on the known information.
Modal logic reasoning	Modalities are concepts like possibility, reasoning about possibility such as probability and impossibility.
Hybrid reasoning	Combines two or more of the above reasoning approaches to provide a robust solution.

Table 3.1 Reasoning Methods [Turban_a et al., 2005 adapted]

Some problem-solving OO paradigms involve producing a series of plans, and sub-plans which are called upon by a software application to provide a solution. Each plan has a specific pre-condition, which is required to be fulfilled prior to the plan's execution, known as a trigger [Turban_d et al., 2005]. The plan can be used to perform an action, send a message, or change the goal to be achieved. The system requires that plans or partial plans are available to complete sub-goals and goals to fulfil the aspired action. The planning decision-making concept is based on finding a solution by connecting together a number of sub-plans.

The fundamental addition to the inference engine concept made by agents is the inclusion of motivational goals, meta-level reasoning about the goal and action choices, which allow the module to act independently of any other application in a self-contained manner. In addition the inclusion of modal logic permits the agent act using probability, when an exact solution is not available (an ability to act in the face of uncertainty).

Message payload

The message payload relates to the actual message which is received. The construction of the payload itself can take many forms such as an XML document, schema, or an object interface. The terms ontology and language are both used to describe message payload construction. From a medical perspective many discipline ontologies exist, which permit specialist groups to exchange information using domain specific language nuances. This can make communication of data between different medical disciplines difficult to achieve.

Message envelope

To deliver a message payload to a specific software module it is necessary to wrap it using a message *envelope*. A message *envelope* consists of a number of key parameters that allows the message sender, receiver and content to be clearly identified during message transmission. The principal addition the agent approach contributes is that it includes social components such as performative (a type of speech act which indicates a performative action by the recipient), ontology encoding, protocols, in-reply-to, reply-with and reply-by. These additional components, which are included within the Foundation for Intelligent Physical Agents (FIPA) communication standards, establish a social fabric in which the agents can socialise [FIPA, 2002; FIPA_a 2003].

Communication facilities

The term *communication facilities* relates to services, which are provided by middleware in order to handle the messages. It primarily utilises the message envelope parameters to exchange messages from one location to another. Agent-orientated applications use the communication facilities to exchange data, but also use the facility to establish a social fabric between agents.

The novel ASK-agent approach of using agent technology to reproduce clinical and laboratory guideline functionality, is not directly supported by any existing agent platform. However, a prototype can be realised by selecting an appropriate platform to provide base-level agent functionality, and building the ASK-agent on top of it. The ASK-agent inference engine must therefore be capable of reproducing functionality found in clinical and laboratory guidelines. So the type of inference execution used by the agent is critical to the ASK-agent design. Section 3.4 investigates a number of agent platforms that are capable of providing the base-level functionality on which the ASK-agent can be built. From a medical perspective the principal aim of data communication is to provide a method of transmitting data that preserves its context when it is exchanged between systems and sub-modules, whereas data communication from an agent perspective must also include a social element.

3.4 Agent-oriented paradigm and implementation examples

In accordance with the evolution of the software timeline as depicted in Figure 3.1, the computer landscape has diversified from standalone computer systems, to distributed open and dynamic systems. To realise distributed systems that can operate within these open and dynamic environments, it has been necessary to increase the operational granularity of software modules and to improve communications and interfacing between the respective software components [FIPA_b, 2003]. This has necessitated the need for some degree of autonomy to be built into component design [Luck et al., 2004] to enable them to respond dynamically to changing circumstances while trying to achieve a particular objective.

The principle behind the agent approach relates to a software module that can act on behalf of a body of knowledge. To appreciate the objective of acting on a body of knowledge's behalf to accomplish a task that was not predefined, the ASK-agent must provide a storage container (i.e. knowledge representation) for the domain knowledge and a control and decision mechanism to act on that knowledge (i.e. decision-making or inference logic). This section will discuss agent frameworks and methodologies to provide these essential components.

The concept of the agent-oriented approach was initially introduced by Bratman in 1987 [Bratman, 1987]. However the idea of *agent-oriented programming* (AOP) was first discussed by Yoav Shoham [Shoham, 1993] in 1993, then further refined using *belief*, *desire* and *intentions* (BDI) and presented as an agent language devised by Rao [Rao et al., 1995] in 1995. Rao suggested that a *belief* captures the *informational* attributes, the *desire* captures *motivational* attributes and the *intention* captures the *deliberative* attributes of an agent, which allow agents act autonomously [Pokahr_b et al., 2005]. Rational agents (a class of agent whose operation is based on reason) have an explicitly described *world-model* representation and achievement objectives. Rationality means that the agent will always perform the most promising actions (based on the knowledge about itself and reasoning engine algorithm) to achieve its objectives. The agent may not know all the effects of an action in advance; therefore it can deliberate over available options to determine the most appropriate action to choose. For example, a game-playing agent may choose between a safe action, and an action, which is risky, but has a higher reward in case of success.

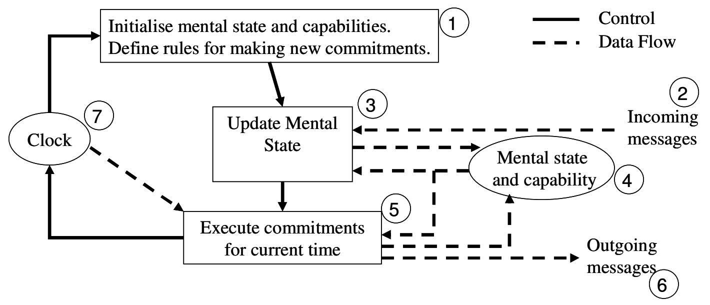
To realise rational agents, numerous deliberative agent architectures exist [AgentLink, 2005]. Three of the most widely used agent architectures are agent-oriented programming (AOP) (section 3.4.1), belief desire intentions (BDI) (section 3.4.2), and knowledge-base, goals and plan (KGP) (section 3.4.3) and they are discussed below.

3.4.1 Agent-Oriented Programming (AOP) model

AOP is based on the societal view of computation, where messages between agents forms an integral part of the agent's operation. The agent is represented using notions of belief, capabilities, choices and commitments [Bordini & Hubner, 2005]. An example of an AOP interpreting language is Agent-O. In Agent-O the agents have capabilities, initial beliefs, initial commitments and commitment rules [Parks, 1997; Shoham, 1993]. Before committing to a course of action, the received messages and the current beliefs are matched against the agent's operating rules. The Agent-O model follows a simple control loop when executing, where during each step it completes the following:

1. Gathers incoming messages and updates the mental state accordingly.
2. Executes commitments using capabilities.

The Agent-O program cycle of control is shown in Figure 3.3. Although the Agent-O program appears to be logic based, nothing like proposition-matching is involved [Parks, 1997]. Proposition logic defines a relationship between the subject and predicate using statements which are either *true* or *false* (e.g. "*men are mortal*" which is *true)*. The process of execution is a straight forward command execution. The capabilities, mental state and commitments in the Agent-O program are all stored in a semantic type network.



Initialise mental state and capabilities. Define rules for making new commitments.
1
Control
Data Flow
2
Incoming messages
3
Update Mental State
7
Clock
Mental state and capability
4
5
Execute commitments for current time
Outgoing messages
6

Figure 3.3 Agent-O language interpreter control/data flow [Shoham, 1993]

Figure 3.3 Agent-O language interpreter control/data flow explanation:

1. The cycle begins by initialising mental states and capabilities (1).
2. The duration of time spent by the agent processing its actions affects the usefulness of its outcome. Therefore, the clock (7) is set to ensure the current operation is processed within a particular period of time.
3. The incoming messages (2) are received and used to update the agent's mental state (3).
4. The activity of updating the mental state triggers the mental state and capability database (i.e. a semantic network) (4) used to select an appropriate activity to commit to.
5. The commitment is then executed (5) by completing the action, or sending an outgoing message (6) and/or resetting the clock (7).
6. New commitment rules are then defined reiterating the cycle of operation (1). The cycle starts at the beginning again.

3.4.2 Belief-Desire-Intention (BDI) model

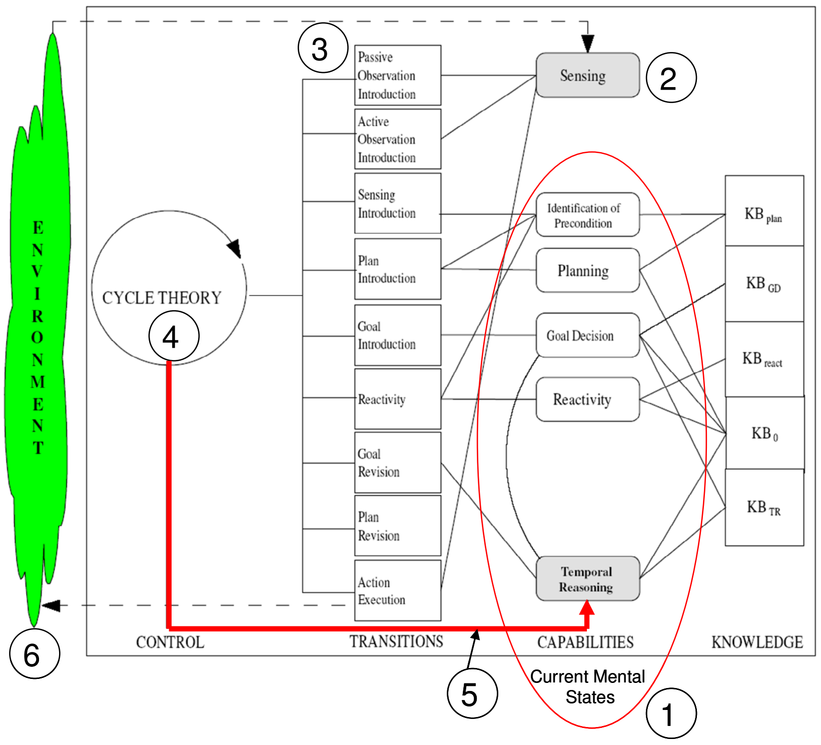
The Belief-Desire-Intention (BDI) architecture expanded on the AOP presented by Shoham, and proposed that an agent must have artificial mental attitudes of belief, desire and intentions (BDI) representing the information, motivation and deliberative states, respectively [Rao et al., 1995]. These attributes determine the agent's behaviour and are critical for achieving adequate or optimal performance when deliberation is subject to limited resources (e.g. time, processing power) [Pokahr_b et al., 2005]. The BDI aspect of the model is primarily based on modal logic, and meta-level reasoning. BDI implementations use an unambiguous formal language with a clear syntax for describing their rational state, and have a high level of social activity [Pokahr_b et al., 2005]. BDI agents permit their knowledge based representation to be accessed and viewed by other agents [Bordini & Hubner, 2005]. Therefore, other social members have some degree of *introspection*[9] into each agent's current beliefs and reasoning at any point in time. Prior to committing to collaboration activities, an agent can inspect the goals and beliefs of the other group members and reason about their attitudes and notions. Some BDI implementations also have a richer knowledge base language allowing obligation, permission, concepts and probabilities to be described. Various BDI architectures exist and four in particular are discussed in more detail later in this section. Recent developments in BDI research have identified that emotion plays an important part in human decision making and a number of agent platforms have began to adapt their execution to reflect this [Pokahr_a, 2005]. Guidelines and protocol documents do not contain any emotional components, so aspects of an agent's operation related to emotion will not be discussed.

3.4.3 KGP (Knowledge-base, Goals and Plan) model of agency

There are obvious similarities between the KGP model of agency and the BDI models given by the similarities between KGP's knowledge, goals and plan and BDI's beliefs, desires and intentions, respectively [Kakas, 2003]. However, the BDI model is based on modal and meta-level logic, and the gap between a BDI specification and its practical implementation encompasses a much wider scope for encoding creativity by the software designers than the more explicit KGP model. KGP is based on procedural/temporal reasoning where the environment is modelled and real observations are compared to it [Kakas, 2003]. This comparison between the environment and the model permits the agent to measure its actual position in relation to its desired position. The operations required to achieve the desired position are controlled by a cycle theory (i.e. the name given to the theory developed to choose operations so the desired position or outcome can

[9] *Introspection* –self-observation and reporting conscious inner thoughts and desires.

be achieved from the actual position). The role of the cycle theory is not to provide fixed control on the operation of the agent, but to regulate it, using the model comparison to obtain a desired pattern of behaviour. Figure 3.4 shows the KGP agent model. The knowledge bases (KB) used in this framework are based on the semantic network format and the nodal tree format.



Legend

Item	**Description**
KB	Knowledge Base
KB_0	Stores dynamic knowledge of the external world including what the agent has observed actions it executed and the communications it received.
KB_{Plan}	Stores partial plans that can be called to achieve goals.
KB_{TR}	Stores the agent's temporal reasoning logic.
KB_{React}	Stores the reaction triggers used to select plans and identify the preconditions for the successful execution of given actions.
KB_{Goals}	Stores the goals used to select plans and identify the preconditions for the successful execution of given actions.

Figure 3.4 KGP agent model [Kakas, 2003]

Figure 3.4 KGP agent model explanation:

1. The operation of the system begins by combining its current mental state (of planning, goals, reactivity, temporal reasoning and identification of preconditions) (1) with the sensing of the environment (2).
2. This activity has a weighted effect on the transitions (3) where goals and plans are introduced. All base-level goals, plans and rules are stored in their respective Knowledge Base (KB) elements. They are assembled as high level goals, plans and rules at the transitions (3) section of the cycle.
3. The cycle theory (4) then compares the current behaviour against the desired pattern of behaviour and adjusts the temporal logic (5).
4. If the action is appropriate it executes the action that acts on the environment (6).

The above summary of the three agency models shows that the ability of an agent to choose a course of action is based on artificially produced mental attitudes. These mental attitudes offer agents and multi-agent systems the ability to provide natural (human-like) decisions, and have a high level of abstraction, which simplifies their understanding [Pokahr_b et al., 2005]. For the purposes of this research the selection of an appropriate base-level agent which has the ability to reproduce both the knowledge and inference logic structures contained in guidelines is critical. The AOP approach gives the appearance it is logic based, but nothing like proposition-matching is involved in its execution [Parks, 1997]. For this reason AOP was not chosen as an appropriate avenue for the ASK-agent, because proposition-matching, modal and meta-level reasoning are important aspects of interpreting clinical and laboratory guidelines. The KGP approach is heavily dependent on using an accurate model to simulate the real world environment [Kakas, 2003], and comparing the real-world inputs to the model in order to produce an outcome. It is not the role of clinical or laboratory guidelines to provide a model, or to give enough information for an accurate model to be formed. For this reason KGP agents were not chosen, because guidelines are an abstraction (or interpretation) of a domain viewed from a condition, disease or organ standpoint, and not an exact description of a domain. The BDI approach is based on the belief capturing the informational attributes, the desire capturing motivational attributes and the intention capturing the deliberative attributes of an agent. These attributes correlate to the structure of clinical and laboratory guidelines as discussed in chapter 2. These agents also employ proposition-matching, modal and meta-level reasoning which also match interpretation aspects of the guideline. Therefore, agents that operate using the BDI agent architecture could support the reproduction of guideline content.

However, not all BDI agents are created equally, and in the following subsections a number of implementation examples have been chosen to demonstrate distinguishing operational and conceptual differences. The agents are Procedural Reasoning System (PRS) (section 3.4.4), AgentSpeak and Jason (section 3.4.5), JADE and Jadex (section 3.4.6), and 3APL (section 3.4.7). The key differences between the systems are the method by which they store knowledge, the format in which they are programmed, and the reasoning methods they use to derive a conclusion or provide an outcome. At the end of this section there is a summary comparison of these underlying agent components.

3.4.4 Procedural Reasoning System (PRS)

Procedural Reasoning System (PRS) relies on procedural knowledge representation that describes how to perform a set of actions in order to fulfil some purpose. For example, a sequence of actions "*download patient result*", "*validate patient result*", "*write result to LIS*", "*send result available notice to result requester*", will achieve the goal "*validate results*". Human realisation of achieving specific goals is procedural in nature [Myer, 1997; Georgeff et al., 1989]. PRS provides an environment in which knowledge of this kind about actions and goals can be readily expressed and executed. It provides a framework based around the BDI reasoning model as an instrument for representing and using an expert's procedural knowledge for achieving goals and tasks. This procedural knowledge is embodied in the form of Acts. An Act is a mapping between some environment and a desired plan.

The PRS agent architecture is formulated around three key data components: the database, the Act Library, and the Intention Structure [Myer, 1997]. The database contains a set of beliefs the agent has about its environment. These may be static or dynamic, depending on whether they represent fixed or temporal information about the domain. Internal beliefs are often termed meta-facts. Beliefs are represented in first order predicate logic using a semantic network format. The Act Library contains the agent's Acts. The plot is a set of actions that defines the procedure (i.e. plan of action) to be followed, and the environment defines a specific set of conditions under which the agent should consider the plot. The Intention Structure is a partially ordered graph of all intentions adopted within the system. Each intention is a *Tuple* of some goal and a collection of Acts. This collection is formed from a combination of some initial Act (the one first adopted) together with a collection of sub-Acts, which have been adopted as a consequence of the initial Act.

In Myer's paper [Myer, 1997] an industrial process was used to show the procedural steps representing the execution cycle of the PRS and this is shown in Figure 3.5.

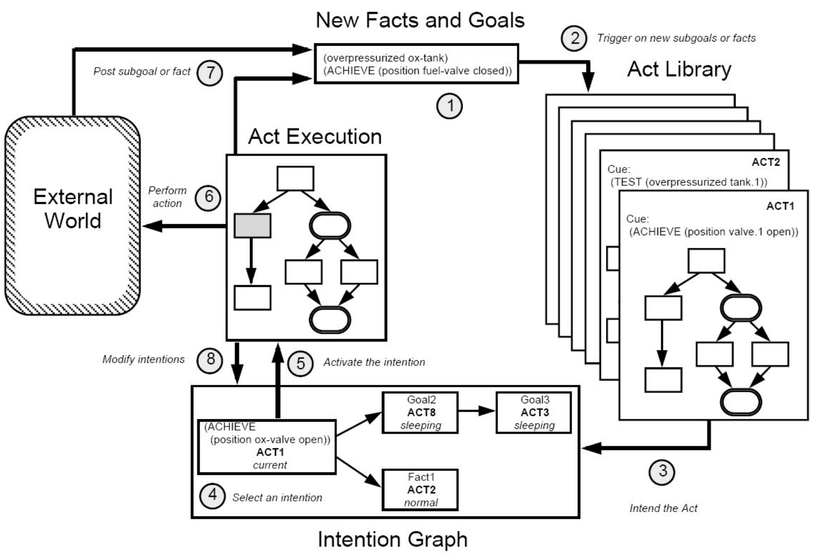


Figure 3.5 PRS Agent Inference Engine [Myer, 1997]

Figure 3.5 PRS Agent Inference Engine explanation:

1. At any particular time, certain goals are established and certain events occur which alter the beliefs held in the system database (1).
2. These changes in the agents goals and beliefs trigger (invoke) various Acts (2). One or more of these applicable Acts will then be chosen and placed on the intention graph (3). Finally PRS selects a task (intention) from the root of the intention graph (4) and executes one step of that task (5).

3. This will result either in the performance of a primitive action in the world (6), the establishment of a new sub-goal or the conclusion of some new belief (7), or a modification to the intention graph itself (8).

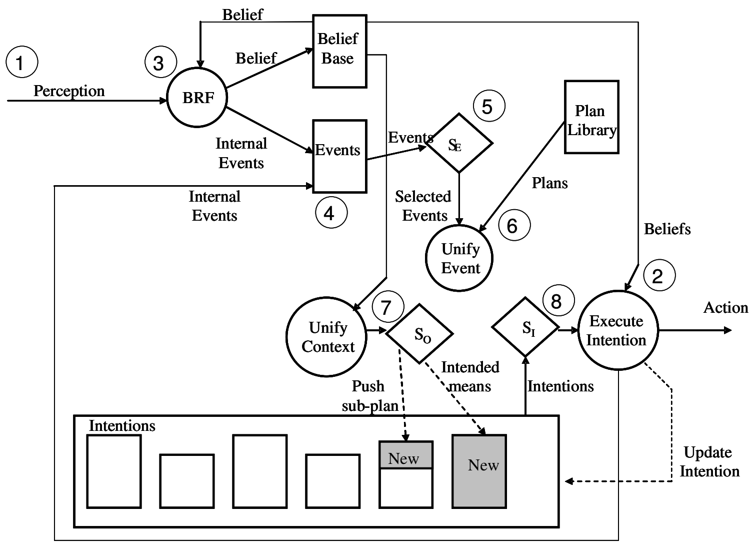
To develop an application using the PRS system the user must supply three separate types of domain knowledge (i.e. facts, acts and intentions), with different files being used for each type of knowledge.

3.4.5 AgentSpeak and Jason

Jason (Java-based agentSpeak interpreter used with Simple Agent Communication Infrastructure (SACI) for multi-agent distribution over the net) is an Open Source extension of AgentSpeak. Jason is an interpreter of AgentSpeak notation and also includes speech-act based inter-agent (formal semantics) communication system. Jason was developed by Rafael Bordini and Jomi Hubner at the University of Durham in the UK and Universidade Regional de Blumenau Brazil. Jason employs SACI to allow the software agents to be distributed over a network. The Jason implementation is based on Java, and therefore offers a multi-platform deployment across a network [Bordini & Hubner, 2005; Hubner et al., 2003].

The AgentSpeak agent is based on modal logic, meta-level reasoning and uses quantitative scoring logic to provide its BDI functionality. An AgentSpeak agent is a reactive planning agent. Plans are triggered by the addition '+' or deletion '-' of beliefs due to perceptions of the environment, or to the addition or deletion of goals as a result of the execution of plans triggered by previous events. AgentSpeak permits two types of goals to be used: achievement goals and test goals. Achievement goals are formed by an atomic formulae prefixed with the '!' operator, while test goals are prefixed with the '?' operator. An achievement goal defines a state of the world where the associated atomic formulae is true. A test goal states that the agent wants to test whether the associated atomic formula is (or can be unified with) one of its beliefs.

Figure 3.6 illustrates the main structure underpinning the operation of the AgentSpeak approach [Bordini & Hubner, 2005]. The rectangles represent sets of beliefs, events, plans, and intentions, diamonds represent selection choices (of one element from a set) and circles represent the processing involved in the interpretation of AgentSpeak programs.



Legend

Item	**Description**
S_I	Intention Selection Function
S_O	Option Selection Function
S_E	Event Selection Function
BRF	Belief Revision Function

Figure 3.6 AgentSpeak and Jason architecture [Bordini & Hubner , 2005]

Figure 3.6 AgentSpeak and Jason architecture explanation:

1. At every interpretation cycle AgentSpeak updates a list of events, which may be generated from perception of the environment (1), or from the execution of intentions (2) when sub-goals are specified in the body of plans.
2. Beliefs are updated from perception. Perception is the process of acquiring, interpreting, selecting, and organising sensory information. This implies the insertion of an event in the set of events. This belief revision function (BRF) (3) is not part of the AgentSpeak interpreter, but rather a necessary component of the agent architecture.
3. The event selection function (S_E) (5) selects an event from the events list (4). AgentSpeak has to unify that event with triggering events in the header section of plans (6). By checking each plans' context in relation to the agent's

beliefs, AgentSpeak can determine a set of applicable plans (plans used at that moment for handling the chosen event).

4. The option selection function (S_O) (7) chooses a single applicable plan from that set, which becomes the intended means for handling that event, and either pushes that plan on the top of an existing intention, or creates a new intention.
5. The intention selection function (S_I) (8) selects one of the agent's intentions (i.e., one of the independent stacks of partially instantiated plans within the set of intentions). On the top of that intention there is a plan, and the formula in the beginning of its body is taken for execution (2).
6. If the intention is to perform a basic action or a test goal, the set of intentions needs to be updated. In the case of a test goal, the belief base will be searched for a belief atom that unifies with the predicate in the test goal. If the search succeeds, further variable instantiation will occur in the partially instantiated plan that contained the test goal (and the test goal itself is removed from the intention from which it was taken). In the case where a basic action is selected, the necessary updating of the set of intentions is simply to remove that action from the intention (the interpreter informs the architecture component responsible for the agent what action is required). When all formulae in the body of a plan has been removed (i.e. has been executed), the whole plan is removed from the intention, and so is the achievement goal that generated it (if that was the case). This ends a cycle of execution, and AgentSpeak starts all over again.

An illustration of the code layout used in Jason is shown in Code 3.1, where the *clinical validation service* is checking a workflow list of results contained in a text file. Each line of the file contains a patient's result, which needs to be validated and is referred to as a slot. The *do* or *perform* instructions are indicated by the symbol "<-".

```
1  // Clinical Validation Service 1
2  // Beliefs
3  // --------------
4  checking(slots).
5  // Plans
6  // --------------
7  +pos(CVS1,X) : checking(slots) & not validating(CVS1)
8  <- next(slot).
9  +validating(CVS1) : checking(slots)
10 <- !stop(check);
11 !markas(slot);
12 !continue(check).
13
14 +!stop(check) : true
15 <- ?valid(X);
16 -checking(slots).
```

Code 3.1 AgentSpeak and Jason code layout

Code 3.1 AgentSpeak and Jason code layout explanation.

Line 2	Locates the *Beliefs* section of the text file.
Line 4	The agent's initial belief is to begin checking the workflow list slots.
Line 5	Locates the *Plans* section of the text file.
Line 7-8	The "+" identifies add plan. When the agent perceives it is in a new location on the list, and not validating it performs the next(slot) command, which moves it to a another slot on the list.
Line 9-12	The plan is used when the agent perceives an un-validated result is present at its current location on the list and its current activity is checking slots. The task of dealing with the perceived result. (1) The agent stops its current activity (line 14-16 then executes). (2) Marks the content of the slot as valid or not-valid, and (3) resumes with the checking slots un-validated result.
Line 14-16	This plan begins with a "+" and a "!". The "!" relates to the must perform plan if the statement is true. The plan states if the agent stops checking for slots (i.e. stop(check)=true), then the agent must compute if the result is valid or not. The "-checking(slots)" is where the agent removes from its belief base checking(slots) goal.

Code 3.1 shows that the Jason agent explicitly details the beliefs and plans using a unique semantic network notation format for storing the knowledge base. This file acts as a central container (knowledge representation) for the belief and plan components of the

agent. The text file is then interpreted through AgentSpeak for execution using the Jason Agent system.

3.4.6 *JADE and Jadex*

JADE (**J**ava **A**gent **DE**velopment) is an open source agent software framework fully implemented in Java, and therefore offers multi-platform deployment across a network. It implements a multi-agent system through a Foundation for Intelligent Physical Agents (FIPA) compliant middleware, with a set of tools that support debugging and deployment [Pokahr_c et al., 2003]. The communication element of the agent offers flexible and efficient messaging, where JADE creates and manages a queue of incoming Agent Communication Language (ACL) messages, which are stored locally and privately by each agent. Each of JADE's components is noticeably separated by design, but fully integrated such as interaction protocols, envelope, ACL, content languages, encoding schemes, ontologies and finally, transport protocols. The agent communication element can currently use the Java Remote Method Invocation (RMI), event-notification, and Internet Inter-ORB Protocol (IIOP).

Jadex is an extended version of JADE that incorporates the BDI deliberation engine. It was developed by Alexander Pokahr and Lars Braubach at the University of Hamburg, Germany [Pokahr_c et al., 2003]. Jadex has the advantage of being able to exploit the comprehensive JADE features described above, with the addition of a BDI deliberation engine. Jadex is an interpreter, which converts user applications into a BDI framework for execution on the JADE platform. Jadex uses an XML structured file called an Agent Definition File (ADF) which allows developers describe and encode the desired operation of the agent. All plans are called using the XML ADF file, but plans themselves are Java classes that extend the core Jadex Plan class [Braubach et al., 2005]. To execute the agent application the XML ADF file and Java plans are converted into JADE platform files, which are hidden from view of the programmer and executed. The programmer can view the operation of their Jadex application using a special inspector facilitator supplied as part of the Jadex platform. The operation of the JADE and Jadex system is illustrated graphically in Figure 3.7.

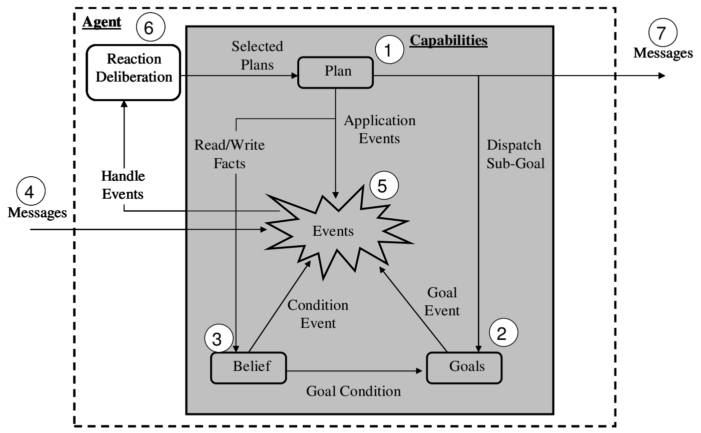


Figure 3.7 JADE and Jadex architecture [Pokahr_b et al., 2005]

Figure 3.7 JADE and Jadex architecture explanation:

The Jadex system is an event driven mechanism where all agent activities trigger events, therefore many activities can trigger simultaneous events that need to be processed by the agent.

1. A Jadex Plan (1) is used to complete basic action, send messages to other agents, but can also update sub-goals and trigger events.
2. Jadex Goals (2) can trigger goal related events.
3. An agent's belief can change during its life cycle resulting in a revision of its beliefbase (3). These changes can trigger a condition event to perform some activity.
4. Incoming messages (4) to the agent are handled as external events within the Jadex system.
5. All events (5) received are prioritised by the Jadex system using quantitative scoring, and where more than one event has equal priority the agent deliberates, using modal logic and meta-level reasoning (6) to select the most appropriate course of action for the agent. The course of action is always a plan, because a plan can send a message, complete a basic task, dispatch sub-goals or trigger events.

An illustration of the code layout used in the Jadex ADF XML file is shown in Code 3.2, where the agent has a belief called "*ALK_P*" and will execute the plan called "*ALK_P_High_Plan*" if the "*ALK_P*" is greater than or equal to 126.

```
1   <agent name="ExpertAgent_ALK_P"
2   package="LiverGuideline.Agent_ALK_P">
3   <imports><import>jadex.runtime.*</import></imports>
4   <capabilities><capability name="dfcap" file="DF" /></capabilities>
5   <beliefs>
6         <belief name="ALK_P"     class= int>
7               <fact>230</fact></belief></beliefs>
8   <plans>
9         <plan name="ALK_P_High_Plan">
10        <body>new ALK_P_High_Plan()</body>
11        <trigger><condition>($beliefbase.ALK_P &gt;= 126)
12  </condition></trigger>
13        </plan>
14  </plans>
15  </agent>
```

Code 3.2 Jadex ADF XML file layout

Code 3.2 Jadex ADF XML file layout explanation.

Line 1	Defines the name of the agent.
Line 2	Defines the name of the package which has an identical meaning to the declarations used in C++ and Java programs.
Line 3	Defines the applications that need to be imported.
Line 4	Defines the agent's capabilities. In this case the agent inherits the standard *dfcap* capabilities.
Line 5-7	Defines the agents beliefs, in this example the agent has one belief called "*ALK_P*" and is set to *230*.
Line 8-10	Defines a plan called "*ALK_P_High_Plan*".
Line 11-13	Defines the condition used to trigger the plan that will activate when the *$beliefbase.ALK_P* element is greater than or equal to 126.

The above explanation is based on Jadex Release 0.932. On the 15th June 2007 Jadex Release 0.96 was launched. Although the latest version included many new features, the main change to the execution cycle was the addition of components to manage agent emotions [Pokahr_a, 2005]. As guidelines do not contain any emotion based components the operation of the legacy 0.932 version has been retained within this thesis when discussing the Jadex platform.

3.4.7 3APL (An Abstract Agent Programming Language)

The 3APL (An Abstract Agent Programming Language "triple-a-p-l") is an agent implementation developed at the University of Utrecht, the Netherlands, by a research team headed by Mehdi Dastani [3APL, 2005]. The 3APL implementation uses the following mental states:

(a) Beliefs:	General and specific information about the environment.
(b) Goals:	Objectives that the agent wants to reach.
(c) Capabilities:	Actions the agent can perform.
(d) Plans:	Procedures to achieve the objectives.
(e) Reasoning rules:	Reason about goals and plans.

(i)	Goal planning rules:	how to achieve goals.
(ii)	Plan revision rules:	how to modify plans.

The BELIEFBASE, CAPABILITIES, GOALBASE, PLANBASE and PG-RULEBASE each play a vital role in the operation of the system. The operation of the 3APL system is illustrated graphically in Figure 3.8.

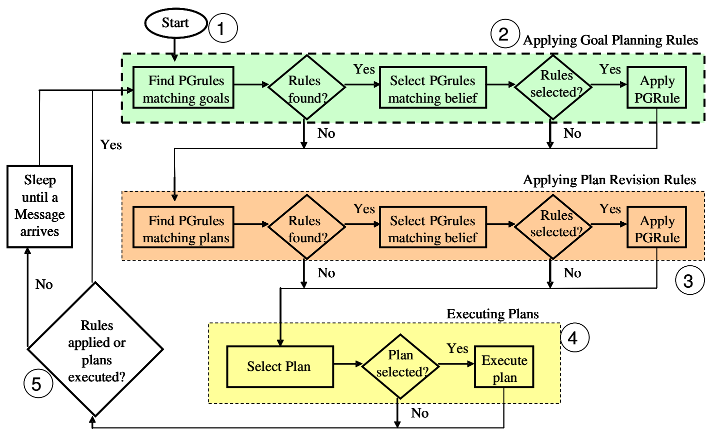

Start
1
2
Applying Goal Planning Rules
Find PGrules matching goals
Rules found?
Yes
Select PGrules matching belief
Rules selected?
Yes
Apply PGRule
No
No
Yes
Sleep until a Message arrives
Applying Plan Revision Rules
Find PGrules matching plans
Rules found?
Yes
Select PGrules matching belief
Rules selected?
Yes
Apply PGRule
No
No
3
No
Executing Plans
4
Rules applied or plans executed?
5
Select Plan
Plan selected?
Yes
Execute plan
No

Figure 3.8 3APL execution cycle

Figure 3.8 3APL execution cycle explanation.

The deliberation cycle of the system begins with a starting instruction or a message arrival (1). The cycle is divided in to three distinct operations.

1. The application of the goal planning rules (searching the 'GOALBASE' for matching goals) (2). The agent searches for PG-Rules which match the current goals of the agent. Of the PG-Rules found the beliefs of each rule are

compared to the agent’s current beliefs. If a PG-Rule is matched, the PG-Rule is applied.

2. The application of the plan revision rules (‘RULEBASE’) is searched and if suitable PG-Rules are found it identifies the beliefs attached to them (3). The agent searches the beliefs of the chosen PG-Rule to match to current beliefs. If a PG-Rule is matched then the PG-Rule is applied.
3. The executing of the plans (4). If the rule is still plausible it is applied, the plan executed, and the system waits for the next activity (5) (e.g. incoming message, or start the cycle again).

The configurations of the BELIEFBASE, CAPABILITIES, GOALBASE, PLANBASE and PG- RULEBASE attributes are accomplished using a single text file, as illustrated in Code 3.3. The *Validating* program is checking a workflow list of results in a text file. Each line of the file contains a patient’s result, which needs to be validated and is referred to as a slot.

```
1  PROGRAM "Validating"
2
3  CAPABILITIES{
4
5      { pos(P) } Goto(R) { NOT pos(P) , pos(R) },
6      { pos(P) AND notChecked(R) } Validate(R) { NOT notChecked (R) },
7   }
8  BELIEFBASE{
9        notChecked (slot 4).
10       destPosition(slot 2).
11       pos(slot 3).
12 }
13 GOALBASE{
14     Validate(), MovePosition()
15 }
16 PLANBASE{ }
17 RULEBASE{
18      Validate()<-notChecked(slot) | BEGIN Goto(slot);Validate(slot) END,
19 }
```

Code 3.3 3APL example code extract

Code 3.3 3APL example code extract explanation:

Line 1	Declares the program name.
Line 3	Locates the CAPABILITIES section of the text file between the {} accolades.

Line 5	Using the capability formula {Pre} Capability {Post} it is interpreted as: When at pos(P) “Goto” pos(R), then remove the belief the agent is at pos(P) from the beliefbase and insert the new position pos(R).
Line 6	Using the capability formula {Pre} Capability {Post} it is interpreted as: When at pos(P) AND result is “notChecked”, then validate the result and mark it as valid or not. Then remove the “notChecked”belief from the beliefbase.
Line 8	Locates the BELIEFBASE section of the text file between the {} accolades.
Line 9-11	Describes the current beliefs of the agent.
	notChecked(slot 4) identifies the workflow list slot as “notChecked”.
	destPosition(slot 2) identifies the next destination slot.
	pos (slot 3) identifies the current slot position.
Line 13	Locates the GOALBASE section of the text file between the {} accolades.
Line 14	Describes goals the agent wants to achieve in order of priority. The first goal indicates the agent wants to validate slots, and the second goal indicates the agent wants to move position.
Line 16	Locates the PLANBASE section of the text file between the {} accolades. In this example there are no plans when the agent is initialised.
Line 17	Locates the RULEBASE section of the text file between the {} accolades.
Line 18	The rules take the form Head <- Guard | Body, where if the head of the rule matches an agent’s goal and the belief base of the agent entails the guard of the rule (rule condition), then the body of the rule, which is a plan, will be added to the plan base.

3.4.8 Summary of agent approaches

A summary comparison of the three key operational aspects of each agent is given in Table 3.2. Each of the four agent platforms examined were agents modelled on the BDI approach.

Aspect	PRS	AgentSpeak & Jason	JADE & Jadex	3APL
Knowledge Base	Uses three different types of knowledge. Act files constructed using a specialised act-editor & graphs. Database files constructed using semantic network format. Function files stored as semantic network text. Files parsed into semantic networks for use by agent during run-time.	Text file using semantic networks format. File is parsed into semantic networks and object knowledge representation for use by the agent during run time.	XML used to describe the knowledge base. XML then parsed into Java objects for use by the agent during run time.	Written in a text file using a standard text editor. The file is parsed into semantic networks for use by the agent during run time.
Format	BDI operation is complex, which requires access to all three knowledge types. The BDI is not represented in one single file for editing. Any changes need to be represented in all three knowledge types.	BDI encoded using a single definition file, which contains the beliefs, plans and goals of the agent. The coding of the file not standardised and used very specific notation (e.g. "!", "?", "+" and "-").	BDI encoded using a single definition file ADF. The coding of the ADF is kept simple by the use of XML tabs.	BDI encoded using a single definition file which contains the beliefs, plans and goals of the agent. The file coding is based on the semantic network format.
Reasoning	The selection of the appropriate plan is based on formal and procedural reasoning.	Plans are prioritised by virtue of their order of appearance in the text document. It uses quantitative scoring which is procedural/temporal reasoning. The selection of the appropriate plan is based on modal reasoning and meta-level reasoning.	Plans are prioritised using quantitative scoring which is procedural/temporal reasoning. The selection of the appropriate plan is based on modal reasoning and meta-level reasoning.	Plans are prioritised by virtue of their order of appearance in the text document. It uses quantitative scoring which is procedural/temporal reasoning. Selection of the appropriate plan is based on modal reasoning and meta-level reasoning.

Table 3.2 Agent modelling language properties

The original BDI agent was the PRS platform, and formed the foundation around which the other three agent platforms (i.e. Jason (AgentSpeak), Jadex (JADE), 3APL) were very loosely developed in order to explore different types of reasoning combinations. To analyse subtle differences between the four systems it is necessary to generalise on the

PRS approach and then summarise the main differences each of the other agent systems has made.

Consider for example the following PRS cycle:

1. Observe environment, agent state and update event queue to reflect observed events.
2. Generate new possible goals (tasks), by finding plans whose trigger matches event queue.
3. Select matching plan for execution (an intended means).
4. Push the intended means onto the intention stack.
5. Select an intention and execute next step of its topmost plan (intended means):
   If the step is an action, perform it.
   If it is a sub-goal, post it on the event queue.

The Jason agent adds the ability to test or achieve a goal at juncture 4 in the cycle, which was not in the original PRS agent. These additional goals allow agents to perform meta-level and modal reasoning about possible future goal states, before committing on a specific course of action. The Jason agent also provides an advanced symbol notation contained in a single text file, which can be used to encode the selection of plans and goals, and a facility to handle plan failures.

The Jadex agent alters the dynamic of the PRS cycle example by allowing all plans, goals, and beliefs an equal opportunity to create events. Every event generated involves a full re-evaluation of the current beliefs, desires and intentions of the agent. Therefore, when the agent is currently active in relation to a goal or sub-goal, this activity is paused if an event of a higher priority has been triggered. The Jadex agent adds the ability to select *perform*, *achieve*, *query*, *maintain* and *meta-goals* and provides the facility to perform modal and meta-level reasoning in order to choose an appropriate response to events. The BDI structure for Jadex is developed in an XML format and the one file contains all the information necessary to define the operation of the agent. All plans are stored in Java.

The cycle of the 3APL agent alters the PRS example by adding the ability to perform meta-level and modal reasoning for the deliberation of goals. The main driving force of the 3APL agent is its ability to describe and support the notion of goals. An abstract plan cannot be executed directly in the sense it updates the belief base of an agent. Abstract plans serve as an abstraction mechanism like procedures in procedural programming.

This section described the operation of an agent from a single agent viewpoint, where the agent stored information, logic and performed reasoning about problems it is trying to solve. It only focused on the operation of a standalone agent and its ability to act autonomously. Indeed the fundamental power behind the agency approach is the ability of each agent to be self-governing and autonomous. However, for the agency approach to truly reproduce the way in which clinical and laboratory guidelines are used, it must provide some means for these predominately loosely coupled modules (i.e. agents acting on behalf of guidelines) to cooperate, collaborate, and negotiate to achieve common goals. Therefore it is vital the communications between agents and other users are carefully constructed and controlled and before selecting a base-level agent it is important to discuss the communication aspect.

3.5 Software communications

The principal goal of medical data communication is to preserve the meaning and understanding of the information when communicated to another application. From a medical perspective there are three dominant medical standards for communications which are: (i) CEN's ENV 13606-4:1999, (ii) the HL7 and to a lesser extent the newly established (iii) *open*EHR. Agent communication has a similar principal goal to that of the medical data, but must also provide a means for the agents to interact at a social and operational level [FIPA_b, 2003].

From an agent technology standpoint there are two agent communication languages with the broadest uptake, and they are KQML and FIPA Agent Communication Language (ACL). KQML was developed in the early 1990s as part of the US government's (Advanced Research Projects Agency) ARPA knowledge sharing project, and is a language and protocol for exchanging information and knowledge [Luck et al., 2004].

FIPA is a non-profit organisation, which produces standards for the interoperation of heterogeneous software agents [FIPA_a, 2003]. The FIPA standard incorporates and extends many KQML aspects, and has been the most widely used agent communication standard to date [Luck et al., 2004]. Both the medical and agent communication languages achieve their exchanges by simultaneously integrating two types of information within a message; (i) lower-level information (message *payload*), and (ii) meta-information about the content of the message (message *envelope*) [FIPA_b, 2003; CEN_a, 1999], and by providing communication facilities in order to handle and process the messages. The principal concept is that the message payload is a structure such as XML or schema, or object interfaces, the contents of which preserve the meaning and understanding of the information. The message *envelope* component relates to how the message is seen at a network level between the message sender and receiver. The communication facilities relate to the handling of messages by the system. All three components are discussed in more detail in a context, which will be used in Chapter 4, to verify if standard agent communications can support standard medical data exchanges.

Message payload

The message payload relates to the actual message that is received. The construction of the payload itself can take many forms. From a CEN pre-standard prEN13606:2004(E) standpoint a particular message payload type is not insisted upon, and formats such as an XML document, schema, or an object interface are all acceptable. A recently employed term used in medical informatics is also described in the CEN pre-standard, that of an *archetype*[10]. An archetype is a re-usable, template model which presents a specific viewpoint of a domain reference model [CEN_e, 2007]. The key feature of the archetype approach is to separate information models, such as object models of software, models of database schemas, from the domain reference model. Unlike some software engineering methods, the archetype is not simply represented by another layer of a class model, but a library of domain concepts, authored in a constraint language. Archetypes have two key purposes: (a) they allow domain experts such as clinicians to create the definitions which

[10] *Archetype* - is the formal definition of prescribed combinations of the building-block classes defined in the *reference model* for particular clinical domains or organisations. An archetype is a formal expression of a distinct, domain-level concept, expressed in the form of constraints on data whose instances conform to some class model, known as a reference model [CEN_e, 2007].

will define the data structuring in their information systems, and (b) they provide a basis for intelligent querying of data. From the agent FIPA standpoint a similar set of payload formats is acceptable [Ingram et al, 2006; Schloeffel, 2003]. The FIPA system accepts any Java Object (e.g. XML, or a customised schema can be instantiated as a type of Java Object).

All of these formats preserve context of data when communicating information from one system to another. XML provides a basic syntax used to share information between different kinds of computers, different applications, and different organisations. In this case the XML file itself is the message payload. The schema approach is like a diagram or plan, where the information elements in the message are stored in a rigid format, and have a relationship to each other by virtue of their position in the schema. The receiving party is aware of the schema structure and can access the information slots to retrieve the required information. In this case the schema data file itself is the message payload. The object approach is where information is transmitted as a software object. The receiving party can interface with the object to retrieve the desired information. In this case the object itself is the message payload. Therefore the payload holds (or contains) the actual context rich medical information to be exchanged between two or more systems. But for communications to work effectively it is vital that the message gets to the correct destination using a message envelope.

Message envelope

A message *envelope* consists of a number of key parameters that allows the message sender, receiver and content to be clearly identified during message transmission, such as *EHCR source, EHCR destination, EHCR message* for the CEN standard, and *FIPA sender, FIPA receiver* for the FIPA agent standard. The envelope data is used to direct the message to the correct location, or ensure it follows a particular workflow. The use of the message envelope is extended using the FIPA standard to allow the agents to socialise and collaborate.

Communication facilities

The term *communication facilities* relates to services, which are provided by middleware in order to handle the messages. It primarily utilises the message envelope parameters to direct the message to the correct location using the computer network. From the CEN standard perspective this is realised through a number of instantiated set routines, such as *provide EHCR message*, *request EHCR message* and *specialist service request message* etc. These routines are developed by the party implementing an instance of the standard.

From the FIPA communication perspective this facility is provided through the base-level agent platform. Figure 3.9 illustrates the FIPA agent management reference model. The agent platform provides the physical infrastructure in which agents can be deployed. In the figure the *Agent* represents a computational process that implements the autonomous, communicating functionality of an application. Agents communicate using an Agent Communication Language (ACL). The Agent is the primary and mandatory actor of the agent platform, which combines one or more service capabilities. An agent must have at least one owner, and an unambiguous identity represented by the Agent Identifier (AID). A *Directory Facilitator* (DF) is an optional component of the agent platform, but if it is present, it must be implemented as a DF service. The DF provides *golden-pages* services to other agents. Agents may register their services with the DF, or query the DF to find out what services are offered by other agents. Multiple DF's may exist within an agent platform and may be federated. The *Agent Management System* (AMS) is a mandatory component of the agent platform. The AMS exerts supervisory control over access to, and use of, the agent platform. Only one AMS will exist in a single AP. The AMS maintains a directory of agent AID's and addresses in the form of *white-pages* for all agents (mandatory) in the system. These *white-pages* are used only if the name of the agent (AID) is known and its address is required. Only the DF service provides facilities to search for service descriptions, names, owners and types. The Message Transport Service (MTS) is the default communication method between agents on different agent platforms.

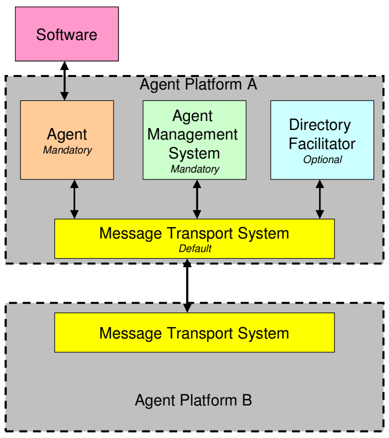


Figure 3.9 FIPA Agent Management Reference Model [FIPA_b, 2003]

Therefore, the communication facilities from the agent perspective are constructed in the platform using the mandatory agent actor. The AMS is employed as part of the platform, and is in existence before any agents are instantiated. The locations of other agents in the platform are stored within *white-pages* directories (AID information) in the AMS. The AMS exerts supervisory control on all communications within the platform and acts as the messaging handling engine. The optional DF component can be set-up to store additional information to suit the specific agent application, where bespoke *golden-page* type data can be stored and searched in order to identify specific agents.

3.6 Software modelling

A modelling language is any artificial language that can be used to express information or knowledge or systems in a structure, which is defined by a consistent set of graphical or textual rules [Padgham et al., 2005]. The rules are used for meaning interpretation of the components in the structure. The development of a system that provides a robust patient-centred validation service (see section 2.2), requires various contributions from a number of different groups. These groups are stakeholders (e.g. doctors, clinicians, managers, technologists and laboratory staff) in the system, and must be able to provide an input and/or comment during its design, implementation and maintenance. It is therefore vital an interface is established that gives these groups an opportunity to communicate their ideas and concepts during the design/implementation process. To overcome these issues a number of software modelling paradigms have been developed. Unified Modelling Language (UML) is one of the most widely used object-oriented modelling languages. However, the operations of agent software are somewhat outside the scope of UML, and a new improved modelling format needs to be developed. Four of the key properties this new paradigm needs to incorporate are listed in Table 3.3 [Padgham et al., 2005].

Property	**Description**
Autonomy:	The ability of an agent to operate without supervision.
Reactiveness:	The ability of an agent to respond in a timely manner to changes in the environment.
Proactiveness:	The ability of an agent to pursue new goals.
Sociality:	The ability of an agent to interact with other agents by sending and receiving messages, routing these messages and understanding them. In addition, sociality refers to the ability of agents to form societies and be part of an organisation.

Table 3.3 Agent modelling properties

Three contending products capable of providing the new modelling format are: Agent Unified Modelling (AUML), Multi-agent Software Engineering (MaSE), and Prometheus Modelling Language

3.6.1 Agent Unified Modelling Languages (AUML)

AUML is a proposed modelling language to manage the development of agent systems [AUML, 2005]. AUML extends UML software modelling notations. UML provides five top-level views: Use-Case View; Logical View; Component View; Concurrency View and Deployment View [AUML, 2005]. These views are presented graphically to illustrate the operation of the system to any potential reader. This makes it easier for users to understand concepts and ideas and comment on them. It uses seven diagram types as detailed below:

1. Use-Case diagrams showing actors and use cases.
2. Class diagrams showing relationships between classes.
3. Object diagrams showing actual objects (instances of classes).
4. State diagrams showing class state changes.
5. Three types of interaction diagram (Sequence diagrams, Collaboration diagrams and Activity diagrams).

6. Component diagrams showing software components.
7. Deployment diagrams showing physical layout.

AUML expands UML by providing the following inclusions [AUML, 2005; Padgham et al., 2005]:

1. Organised special agent class.
2. New concept of role (an abstract representation of an agent's function, service, or identification within a group).
3. New Agent Interaction Protocol Diagrams.

3.6.2 Multi-agent Software Engineering (MaSE) language

MaSE incorporates analysis, design and implementation using a series of graphically based models with a logical approach to transforming one model into the next [Padgham et al., 2005]. MaSE supports development of heterogeneous multi-agent systems through languages, architectures and environments. The current status of the systems development is quite advanced with diagram tools and translators being available.

3.6.3 Prometheus Modelling Language

The aim of the Prometheus language is to provide detailed guidance to the design team who use agents [Padgham et al., 2005; FIPA_d, 2003]. The language supports all phases of development (such as coding, testing and maintenance), and facilitates tool support and automation. The Prometheus paradigm provides design artefacts to describe a system from overview to detailed implementation, and uses existing approach/standards where appropriate. The current status of the system is in book form describing specification and design, with a prototype tool supporting the design process [Padgham et al., 2005]. A manual translator has been developed in Prometheus to provide a Java Development Environment (JDE) plug-in and produces skeleton code, which is compatible with some agent software applications.

In Table 3.4 a comparison of the three agent models to the key desired properties of a new agent paradigm are presented. It is important to emphasise that each property is provided/implemented in various different forms, but none of the systems has identified sociality in a formal way. Agent-Oriented Programming (AOP) was based on the societal view of computation, which makes it ironic that the sociality aspect has been relatively unsuccessful (see Table 3.4). This could be in part due to the complexities of various types of social environments such as: individual, broadcasting, lecturing, meetings, interacting and collaborating. Each of these social types of communication varies in their use and delivery makeup.

Properties	**AUML**	**MaSE**	**Prometheus**
Autonomy	Expressed within the agent class	Expressed by the fact that the role encapsulates its functionality.	Expressed using goals and multiple ways to achieve them.
Reactiveness	Expressed by the set of behavioural diagrams.	Not expressed explicitly. There is no explicit connection between the event and the action taken. Yet, reactiveness can be expressed using conversation state machines	Expressed by reactions to precepts.
Proactiveness	Expressed by the set of behavioural diagrams.	Expressed by the role's tasks. These tasks are modelled using finite state automation	Is expressed by goals.
Sociality	There is no special treatment of this aspect. No formal grouping of societies.	There is no mentioning of the social aspects of a system except for basic communication. No formal grouping of societies.	Does not address organisational structure, beyond interactions between agents. No formal grouping of societies.

Table 3.4 Agent model comparison

3.7 Collaborative systems

Collaboration relates to systems that work together to achieve objectives a single system could not accomplish when working alone [Schumacher, 2001]. The underlying use of collaborative domain knowledge being utilised by the ASK-agent can be divided into two categories:

1. Similar domain knowledge.
2. Overlapping domain knowledge.

Category (1) focuses on two or more applications that use the same core data, but different inference engines to derive a result. Consider for example VALAB, which is a rule-based system for clinical laboratory results, and LabRespond, which is a statistical analysis system for clinical laboratory results [Oosterhuis et al., 2000; Valdiguie et al., 1992]. These packages access patient information stored in the LIS and validate similar types of laboratory results, but use different methods of inference in order to derive a result. They do not communicate with each other, and are effectively *islands of information*. So what benefit can be gained by allowing these packages to collaborate? If both systems report similar conclusions while observing the same domain, but use different modus operandi to derive the result, it provides corroborating evidence whether or not the result is accurate and true. However, the difficulties associated with providing this facility relate to cost and effort. The cost of purchasing and maintaining two or more similar products to provide the comparative information can be excessive to an already stretched laboratory budget, and the effort to ensure that the systems connect and collaborate to produce useful outcomes can be unwieldy. In this collaboration of domain knowledge category each ASK-agent represents a system such as VALAB and LabRespond by providing similar intra-software processing ability, but extends them to include a social fabric to provide inter-ASK-agent information exchanges.

Category (2) focuses on observing different viewpoints of the same domain and using collaborating information in order to derive a result. Consider for example the case of a patient who is complaining of chest pain combined with classic heart related symptoms, such as shortness of breath, tiredness, fatigue, confusion and impaired thinking. The heart is referred to in many clinical and laboratory guidelines, but for this example the heart related issues are narrowed down to heart disease (e.g. Hypertensive, Hereditary or Coronary) as shown in Figure 3.10. Each of the three clinical guidelines represents a different viewpoint of the organ, which is symbolised by the magnifying glasses. By observing the heart through these viewpoints, and allowing supportive information message passing between the viewpoints to occur, an increased understanding of the heart by the overall system begins to emerge. Information considered important by one viewpoint can be shared with any other viewpoint and used as part of their evaluation of the heart. This collaboration information can be used by the various viewpoints in order to derive a result or support a conclusion. It is not necessary that all the viewpoints actually confirm the derived result, because they may not be able to by virtue of their focus on the domain, but they can support the deriving of a result by another viewpoint. These communications help to yield the best supported data from all available viewpoints. In this case the ASK-agent represents a viewpoint expressed through a clinical guideline.

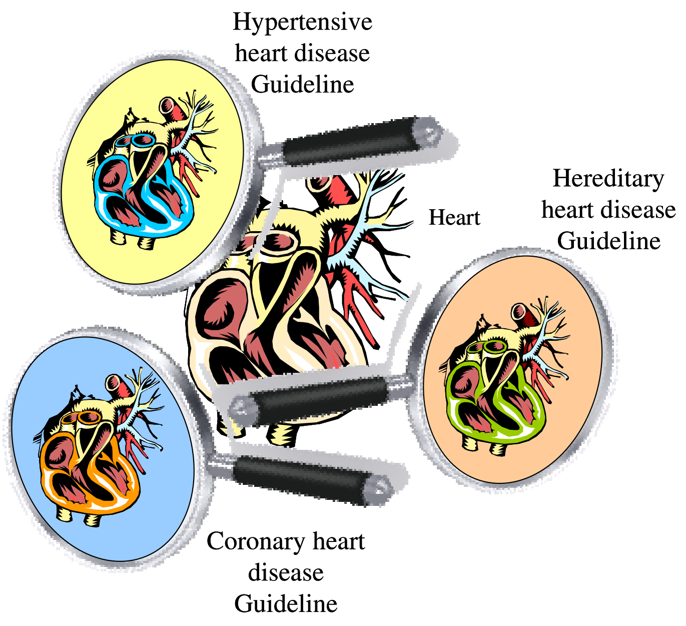

Hypertensive heart disease Guideline
Heart
Hereditary heart disease Guideline
Coronary heart disease Guideline

Figure 3.10 Knowledge overlap

According to Corkill, there are six key challenges involved in creating effective collaborating-software systems [Corkill, 2003]:

1. Representation – getting software modules to understand one another.
2. Awareness – making modules aware when something relevant to them occurs.
3. Investigation – helping modules to quickly find information relating to their current activities.
4. Interaction – creating modules that are able to use the concurrent work of others while working on a shared task.
5. Integration – combining results produced by other modules.
6. Coordination – getting modules to focus their activities on the right things at the right time.

The aim of the ASK-agent is to capture guideline knowledge and logic within a standalone, autonomous module and to provide the ability for *patient-centred* laboratory validation to take place using these modules. Therefore, if an ASK-agent can act on behalf of a guideline, it must be able to collaborate with other agents to provide the *patient-centred* laboratory validation service. But the concept of coordination needs to be further refined with respect to the agent paradigm. Approaches to the coordination of multi-agent systems (MAS) have been recently classified as subjective (typically coming from the distributed artificial intelligence (DAI) community), and objective (emanating from the community of Coordination Models and Languages). Subjective and objective coordination approaches both have very different impacts on the engineering of social aspects of MAS, in particular with respect to the ability to specify and enact social laws to achieve global coherent behaviours [Ricci et al., 2002; Schumacher, 2001]:

1. **Objective coordination**

   The MAS is built by objective dependencies, which refer to inter-agent reliance, namely the configuration of the system in terms of basic interaction means, agent generation/destruction, and organisation of the environment. The management of these dependencies is objective coordination, because these dependencies are external to the agents.

2. **Subjective coordination**

   The agent has subjective dependencies which refer to intra-agent reliance towards other agents. Subjective coordination is the management of these subjective dependencies.

Subjective coordination is dependent on objective coordination, because the first is based on, and supposes the existence of the second. The mechanisms engaged to ensure subjective coordination must indeed have access to the mechanisms for objective coordination [Ricci et al., 2002]. If this is not the case no subjective coordination is possible at all. This does not mean objective coordination belongs to the intra-agent view, but only access mechanisms have a subjective expression in the agent. This is essentially the case for mechanisms like sending or receiving information.

However, a caveat when observing Corkill's six key challenges involved in creating effective collaborating-software is they are simply that, 'challenges', and they do not give a formatted structure or framework on how to provide the collaborating-software. In the following subsections the structures or frameworks of directly connected systems and blackboard systems are discussed for the purpose of allowing ASK-agents to collaborate together.

3.7.1 Directly connected (coupled) systems

A traditional approach to combine a set of diverse software modules together is to connect them according to their data-flow requirements [Corkill, 2003]. In Figure 3.11 a group of six expert knowledge (EK) modules are shown linked together. Each EK module contains local storage (for private attributes), a knowledge source (i.e. algorithm of execution) and a control switch (to link to the other software modules). Each EK module has a choice to contact another EK module that is written into its operation at compile-time. If the EK module follows the true concept of modularity it may appear multiple times in the communication graph, but the connections are predetermined and direct. This design operates satisfactorily when the module sets are static with fixed communications, and coordinated control switching. However, when specific modules are subject to change, and/or when the ordering of modules cannot be determined until specific data values become known at execution time, it makes them difficult to maintain. In addition EK modules cannot be added/removed or altered easily. This inflexibility and lack of control and awareness between the packages hinders them from collaborating to solve problems.

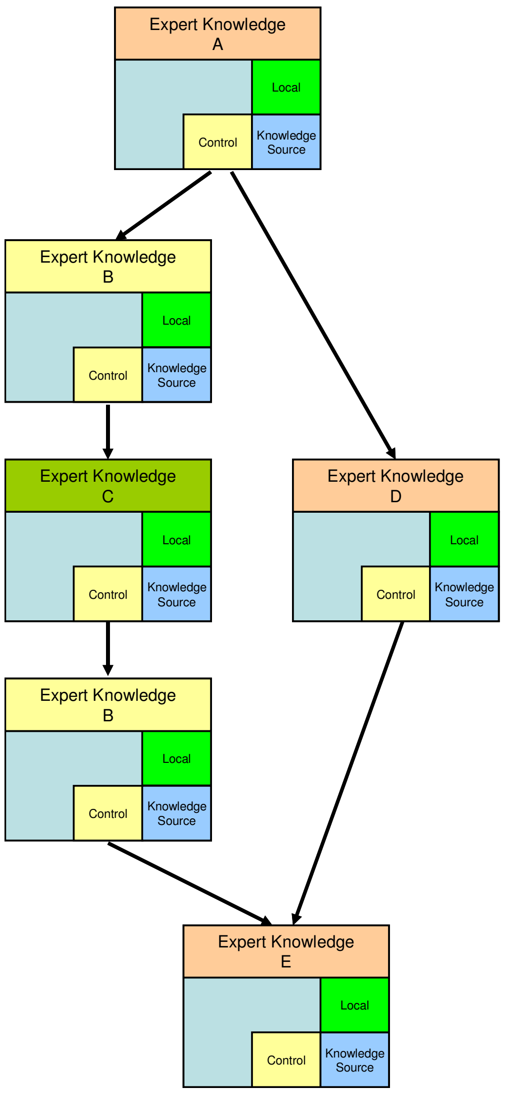


Figure 3.11 Direct coupling

Figure 3.11 Direct coupling explanation:

This figure illustrates the separate expert knowledge (EK) modules A, B, C, D, and E. Each EK has a local control element, knowledge source and local data. The EK modules operate by switching control to another module during their execution cycle. The relationships of all these links are known at compile time. In this example EK module A divides the strand of control to EK module B and EK module D. EK module D operates simultaneously with EK module B, then EK module C, the EK module B again. Then both strands of control merge together ending at EK module E.

3.7.2 Blackboard systems

An alternative to the traditional *directly connected* approach is to automate decision support between various software packages by allowing each system to work independently, and link them together, when necessary, via a blackboard management engine [Corkill, 2003; Lander et al., 1996] as shown in Figure 3.12. The link could be in the form of message passing from one system to another, or simply shared data location in the LIS.

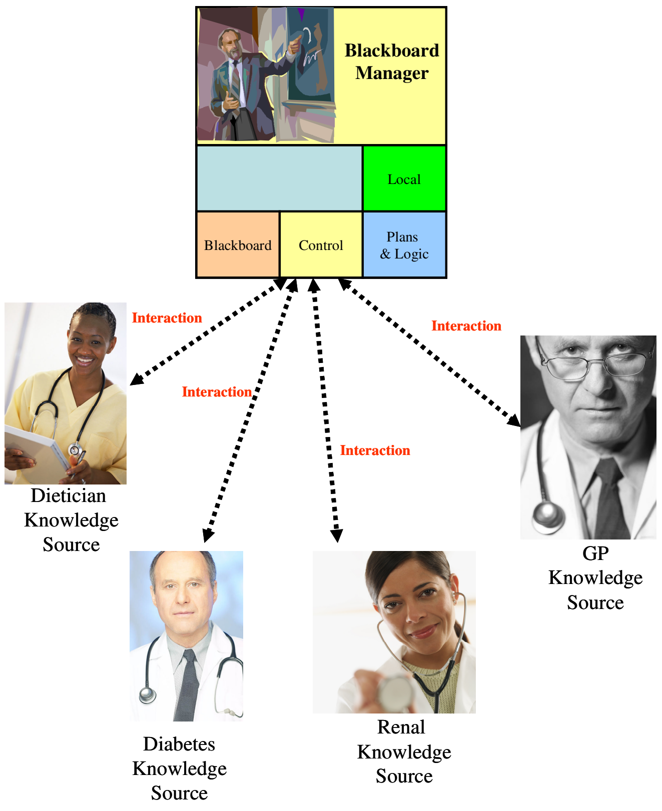


Figure 3.12 Blackboard Management Communications.

Figure 3.12 Blackboard Management Communications explanation:

The diagram illustrates the concept of the centralised blackboard, where common information is shown to all appropriate members of the group. However, the blackboard engine has the task of switching control and interacting with the respective group members. This is fundamentally different from the directly connected system, where the currently active member had the capability to switch control.

The design of a blackboard system is essentially divided into three main components [Corkill, 2003]:

1. **Knowledge Source**

   Knowledge Source's (KS) are independent modules capable of processing information relating to their domain of expertise. KS's do not communicate directly with one another, only through the data on the blackboard.

2. **The Blackboard**

   The blackboard is a communal repository for data, partial plan solutions and other problem solving instruments. The blackboard includes a knowledge based triggering mechanism, which is used to trigger events when certain knowledge based patterns occur. However, the inference provided by the blackboard may not suit all KS's equally, and the system is tightly coupled.

3. **Control Component**

   The control component is responsible for making runtime decisions and inference about the course of problem solving. The control component retains control of the collaboration by selecting the most promising plans from the respective KS's. Difficulty arises when levels of action are required within the control component which in human terms would be actions, such as the pitch of voice or the tapping of the board to extenuate a stress on an element of data or a direction of action to take.

The blackboard has an execution bottleneck, where all KS's must act through the blackboard engine, because KS's have no awareness of each other, and cannot work together as a unit outside the blackboard engine. In general, there are two types of communication actions used for achieving interaction: Pull or Push. Pull communication is when the KS's requests information and only then is the information returned to the requester. An example of this is the web or client-server, only the information the KS requested is returned. Push is when the communication is sent to one or many possible recipients without being specifically requested. An example of this would be radio broadcasting or general e-mail, the information is issued irrespective of being asked.

This blackboard collaboration approach has defined static outcomes for specific choices made during its execution. “Collaboration” software is different from “collaborative” software with the latter being executed dynamically where the end result is not truly predictable in advance [Corkill, 2003; Cockburn 1999; Corkill, 1989]. Mechanisms for validating these software systems can utilise data with known outcomes to ensure the system operates correctly, but the actual inter-software derived outcome is produced *on the fly*.

The manner in which the ASK-agent themselves interact with each other is important and can be considered either passive or interactive. The ASK-agent can simply accept the information passively or interact with the process. The advantages of interactive communication are that it provides:

1. Opportunities for feedback;
2. Increases confidence the message has been understood;
3. Recipients have control over the pace of the communication;
4. Tailoring of messages using ontologies to better meet the recipient’s needs.

The blackboard would be considered a push interactive system, where the desired collaboration information is placed on a blackboard and the individual experts interact with it. The ASK-agent paradigm is considered a push interactive system, where the push is achieved through both broadcasting and direct-message transmissions from an agent with information it needs to share and make public. The way in which the receiving ASK-agent use and apply the information is incorporated into their BDI implementation. But the ASK-agent paradigm is also a pull system, where an ASK-agent can request information and then deliberate on the result. Combining pull and push operation is akin to collaborating software [Hermans, 1996] where the software is involved in the integration and coordination of relatively independent, self-contained modules, which are able to work together effectively on their own [Corkill, 2003].

3.8 *Alternative collaborative approaches*

As a consequence of the agent architecture simulating human decision-making and reasoning, albeit at a high level of abstraction of real human reasoning, it is important to compare agent sociality to human sociality. From an inter-agent social and interactive perspective there are a number of theories which have been developed through the social sciences. Unfortunately, these theories do not directly transpose to software components, but they can act as useful tools to aid in analysing collaboration between standalone entities. The rationale for introducing these alternative collaborative approaches is to provide a focus on the collaborative nature of separate autonomous systems such as individuals, on which agents are based and have the capability to complete certain tasks. If ASK-agents are going to cooperate together as a group then the alternative human based concepts of managing the collaboration need to be considered in the design. For an ASK-agent to truly collaborate, it needs to be able to pursue activities that may not lead to the goal of result validation, but through its action support the validation of the result by other agents. Some of the main stream collaborative theories are given in Table 3.5 [Pokahr_a, 2005; Norling, 2004; Engestrom, 1999].

Title	**Description**
Folk psychology	Provides a means of making choices by incorporating the behaviour of other group members via their reasoning and emotional state.
Connectionism theory	Decision ability offered through group activity fundamentally based on neural networks not rules.
Participation theory	Develops strategic choices based on the act of individuals sharing in the group activities.
Activity theory	Predominately focuses on intentionality, mediation, collaboration and development of constructing objectivity and self-awareness which extends to communities.

Table 3.5 Alternative collaborative approaches

Although the BDI model is primarily based upon a folk-psychological view of reasoning, that is, *the way people think that they think*, as opposed to the actual mechanism of the brain, it is developed at a much higher level of abstraction [Norling, 2004]. The classic folk-psychological view of human behaviour includes many more subtle nuances of introspection and a human emotional stance which are eliminated in the BDI model in order to make implementation simpler [Pokahr_a, 2005]. Although in recent years the agent community has been reviewing the BDI model with a view to developing introspection and emotional components, emotion is not present in guideline documentation, and so it would be of little utility in examining it in this domain.

The connectionism theory is based on developing a neural based network between group members that is used to choose appropriate plans as the neurons fire. The disadvantage of this approach is that an inter-agent communication network needs to be established prior to the act of collaboration, which is simply not present within guideline documentation. In addition to the lack of an established inter-agent network, this theory also reduces the autonomous capability of the agent modules.

The participation theory is based on including all individuals in the process of choosing an activity, which is akin to general election voting, where all individuals have an equal standing in the decision process. This theory is not suitable for use in guidelines collaboration, because each guideline has different levels of importance which affect the choice of activity. These levels of importance relate to both an intra-guideline activity where workflow path selection has levels of priority, and inter-guideline activity where the guidelines themselves have levels of priority regarding their selection.

Finally, activity theory (AT) is based on the classic separation of action from activity [Engestrom, 1999; Leont'ev 1981]. This supports the pursuit of activities that may not lead to the goal of result validation directly, but through its action support the validation of the result by other agents. For example, an action by a validation agent, which requires a blood glucose result in order to validate another laboratory result by the same patient, could use a recent and acceptable urine analysis result available from another agent. This would permit the validation procedure to continue, using this additional knowledge artefact. In addition to the ability to manage indirect group goals the fundamental instruments used in AT have many similarities to the inner-structure of guidelines and the management of collaborating groups of guidelines. These similarities are object-orientedness (not limited to the physical, chemical and biological properties of guidelines, but includes perspective properties which are gained from the separate guideline viewpoints of the domain), hierarchical activity (both *within* and *between* individual guidelines) and mediation tools (which shape the interfaces between different types of guidelines).

Engestrom elaborated graphically Leont'ev ideas by combining numerous models and developed the structure of an activity system as shown in Figure 3.13, which offers considerable structural similarities to the process of collaborating guidelines.

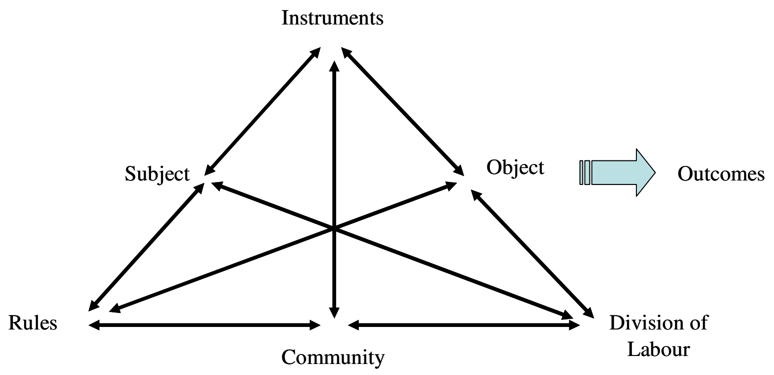


Figure 3.13 Structure of an activity system

This structure of an activity system is based on interlinking components, such as rules, community, subject, object, division of labour and instruments. From a medical perspective this structure can be interpreted as the subject relating to an ASK-agent, which is affected, applies rules to operate, and divides the labour into manageable tasks. The ASK-agent uses instruments (i.e. communication of a message or set of known beliefs, or action) to act on the object (i.e. organ, disease or medical condition of the patient), which could be to achieve some outcome, such as making the patient better. The community of ASK-agents influences the interface (i.e. choice of rules, the division of labour, and the design of the instruments) that the subject can use, but does not directly act on the object or subject itself.

3.9 Discussion

The principles used by modern clinical laboratories for validating numerical results have become a fusion of both medical and software concepts. This chapter examined the concept of artificial intelligence which, for this thesis, focuses on expert decision-making and reasoning in order to replicate the actions and decision of a clinician or a laboratory technologist. It showed that intelligence in modern software applications encompasses two key abilities: firstly the ability of software applications to provide rational processing (i.e. internal inference), and secondly its ability to effectively converse with its community.

This chapter demonstrated the potential of the agent approach to achieve rational processing intelligence, which is realised by their ability to provide a knowledge representation, and the processing of inference logic. Using a BDI agent it was shown that the belief captures the informational attributes, the desire captures motivational attributes, and the intention captures the deliberative attributes of the rational processing intelligence. This synergy between the composition of both the agent and true construction of guidelines as discussed in Chapter 2 suggests that the agents approach is better suited to represent guidelines than the object-oriented approach. This chapter also demonstrated the similarities between agent and medical communications, thereby allowing agents the opportunity to collaborate with other agents and still comply with medical standards and agency sociality.

The final concept of the agent approach discussed was the ability of the agent to act autonomously and collaborate as part of a larger community. Through the use of AT instruments and tools it was shown that an agent or guideline does not need to explicitly describe its role in the larger overall system, in order to operate successfully within the overall scheme. They are standalone autonomous elements focused on a specific universe of discourse, which just happen to be participants in a larger community (i.e. a community of ASK-agents, which focus on viewpoints of the human body). It is the domain (i.e. the human body) itself that binds the ASK-agent to the larger picture allowing collaboration through loosely coupled activities.

Chapter

4

Patient-centred validation service analysis

4.1 Introduction

The aim of this chapter is to collectively analyse the medical laboratory and software issues involved in providing clinical validation from a *patient-centred* standpoint, in order to develop the functional requirements for the ASK-agent architecture. The ASK-agent is based on utilising an existing agent platform and building on top of it the additional functionality to allow it act on behalf of guidelines. Section 4.2 analyses generic guideline content, and explains how they can be reproduced using the ASK-agent. Section 4.3 expands on the generic guideline content by introducing existing lower level CIG functionality components and developing the ASK-agent in order to reproduce this functionality. Both these sections will confirm that an ASK-agent is capable of representing a guideline from a technical content perspective. Section 4.4 focuses on medical communications and demonstrates that the FIPA agent communication standard can be adapted to support medical transmissions among a group of agents, in addition to the social and collaborative activities of the ASK-agent itself. Section 4.5 examines the concept of guideline networking, and remodels a base-level agent platform, in order to provide tools to support heightened social and collaborative activity for the ASK-agent. This novel approach will advance the underlying state of the art agent platform to include the components necessary to allow collaboration.

In parallel to software attempts to provide guideline collaboration activity using agents, it is important to reflect on other disciplines that also apply collaboration techniques. Activity theory stems from the social sciences and focuses on the activity of an individual, as a standalone entity and as part of a group. Although AT does not transpose directly to any software, it does provide insightful collaboration analysis tools and instruments that are examined in section 4.6. The AT concept is further expanded in section 4.7 where it is used to demonstrate that a centralised control engine is not necessary when using the ASK-agent approach.

The base-level agent functionality requirements list is developed in section 4.8. The five agent platforms discussed in Chapter 3 are compared against the list in order to select the most appropriate base-level agent for the application domain. To realise the ASK-agent the base-level agent needs to be expanded, using three customised and mandatory components, which are discussed in section 4.9. Section 4.10 and Appendix C develop the ASK-agent further by identifying principles, operations and methodologies which reoccur in guidelines, and develop agency solutions so they can be reused using the ASK-agent approach.

Finally section 4.11 presents a discussion on the ASK-agents analysis and presents various concluding remarks.

4.2 Guideline components reproduced using agent technology

To illustrate the novel link between guidelines and ASK-agents the following example will address three key components contained in guidelines, and that the ASK-agents approach will need to reproduce: the '*goal task*', the '*decision task*' and the '*action task*'. It will show how these elements are implemented by the ASK-agents proposed in this research. All clinical and laboratory guidelines have these components, but use them within descriptive text and not explicitly in the form of computer logic. This example builds on the foundation work developed as part of the Guideline Interchange Format (GLIF) 3.5 project. Although GLIF is not the only CIG model available, its technical specification was finalised in May 2004 by InterMed Collaboratory, and is regarded as a comprehensive model that includes many of the latest CIG developments [Peleg_d et al., 2003]. It is for this reason it is used as a benchmark for the ASK-agent. GLIF was based on developing a CIG using the object-oriented approach [Peleg_c et al., 2004]. In addition to the development of a medical ontology as discussed in chapter 2, the GLIF project tackled the problem of descriptive text guidelines. GLIF produced a technique by which the '*goal task*', the '*decision task*' and the '*action task*' components could be extracted from the text and encoded into an OO based application. This example builds on the Cholesterol measurement, which was discussed in section 2.7 and Appendix D in relation to its usefulness in detecting liver disease. In these sections the terms such as high density lipoprotein (HDL) and low density lipoprotein (LDL) are defined.

The following text was extracted from the "*National Cholesterol Education Program*" NCEP text-based guideline, and focuses specifically on the Adult Panel III, '*overdue* *cholesterol*' and '*LDL goal*' [NHLBI_b, 2004]. The actual guideline text spans 284 pages, but is summarised in Table 4.1 for the purposes of this example. This summary focuses on the '*goal task*', the '*decision task*' and the '*action task*' components of the '*overdue cholesterol*' and '*LDL goal*' aspects of the guideline.

1 *Check to see if cholesterol test is overdue*
2 If history of heart disease or equivalent and last cholesterol test was over one year ago or if no
3 heart disease and last cholesterol test was over 5 years ago recommend cholesterol test.
4
5 *Determine and establish LDL goal*
6 Identify presence of clinical atherosclerotic disease that confers high risk for coronary heart
7 disease (CHD) events (CHD risk equivalent):
8
9 *Clinical CHD*
10 Symptomatic carotid artery disease
11 Peripheral arterial disease
12 Abdominal aortic aneurysm
13 Diabetes
14
15 *Determine presence of major risk factors (other than LDL) that modify LDL goal:*
16 Cigarette smoking
17 Hypertension (BP 140/90 mmHg or on antihypertensive medication)
18 Low HDL cholesterol (<40 mg/dl)
19 Family history of premature CHD (CHD in male first degree relative <55 years; CHD in
20 female first degree relative <65 years)
21 Age (men 45 years; women 55 years)
22 HDL > 60 acts as a "negative risk factor"
23
24 *Determine LDL Goal*
25 CHD or CHD Risk Equivalents <100 mg/dL
26 2+ Risk Factors <130 mg/dL
27 0-1 Risk Factor <160 mg/dL

Table 4.1 Summarised content of National Cholesterol Education Program guideline

Table 4.1 Summarised sample content of the National Cholesterol Education Program guideline explanation:

Line 1-4 The context of this extract is to examine if a cholesterol test is overdue on a patient. If a patient's history includes any of the following, this guideline recommends performing a cholesterol test on the patient:

- Heart disease or equivalent is present and the last cholesterol test was over one calendar year ago, or
- If no heart disease is present and the last cholesterol test was over 5 years ago.

Line 5-7	Establishes the LDL goal that is to identify the presence of clinical atherosclerotic disease, which confers high risk for coronary heart disease (CHD) events.
Line 9-13	Declares factors which would confirm clinical CHD, such as symptomatic carotid artery disease, peripheral arterial disease, abdominal aortic aneurysm and diabetes.
Line 15-22	Declares features which confirm the presence of other major risk factors (other than LDL) which modify the LDL goal such as cigarette smoking, hypertension (BP 140/90 mmHg or on antihypertensive medication), low HDL cholesterol (<40 mg/dl), family history of premature CHD (CHD in male first degree relative <55 years or CHD in female first degree relative <65 years), Age (men 45 years; women 55 years) and HDL > 60 acts as a "negative risk factor".
Line 24-27	Establishes the LDL Goal, such as CHD or CHD risk equivalents <100 mg/dL, two or more risk factors <130 mg/dL, 0-1 Risk Factor <160 mg/dL.

The conversion of the summarised National Cholesterol Education Program into the equivalent Arden Syntax Medical Logic Module (MLM) is achieved by separating the guideline extract into ‘*evoke*’, ‘*data*’, ‘*logic*’ and ‘*action*’ slots which are shown in Table 4.2. The Arden Syntax (see section 2.6) is perhaps the best-known language for representing clinical knowledge in patient specific decision-support systems and is not unique to GLIF [Peleg_c et al., 2004]. It is a rule-based formalism developed for encoding individual clinical rules as Medical Logic Modules (MLM). The medical knowledge is shared in the form of individual MLM’s, each of which represents a single rule or action. The ‘*evoke*’ slot trigger defines the triggering mechanism for a rule, such as the event of a patient admission. A ‘*data*’ slot defines the information needed to execute the rule. A ‘*logic*’ slot defines the algorithm of execution. Finally the ‘*action*’ slot determines the resultant action to take place.

```
1    Evoke Slot:
2    Upon_Admission
3
4    Data Slot:
5    Cholesterol_Test_UptoDate := read {local (unique to system) SQL call for the
6    following patient variables, 'has_Clinical_CHD',
7    'has_Symptomatic_Carotid_Artery_Disease', 'has_Peripheral_Arterial_disease',
8    'has_Diabetes', and 'Date_Last_Cholesterol_Test'}
9
10   Logic Slot:
11   If ('Clinical_CHD' = true OR 'Symptomatic_Carotid_Artery_Disease'= true OR
12   'Peripheral_Arterial_disease' = true) AND ('Current_Date' -
13   'Date_Last_Cholesterol_Test' >= 1 year) then
14   Alert_Text := "Consider ordering cholesterol profile test because of history of
15   heart disease (or equivalent) and since last cholesterol test was > one year
16   ago.";
17   conclude true;
18   Elseif ('Current_Date' -'Date_Last_Cholesterol_Test') >= 5 years then
19   Alert_Text := "Consider ordering cholesterol profile test because last
20   cholesterol test was > five year ago.";
21   conclude true;
22   else conclude false;
23   Endif;
24
25   Action Slot:
26   Write to Screen Alert_Text;
```

Table 4.2 Equivalent Medical Logic Modules (MLM's).

Table 4.2 Equivalent Medical Logic Modules (MLM's) explanation:

Line 1-2	Declares the evoke slot, which details the circumstances when the MLM will be started. The evoke slot is triggered in this example by the writing into a database of a patient admission to the hospital.
Line 4-8	Declares the data needed to begin and execute the MLM. For GLIF this is an SQL query to obtain the information. Once evoked the system queries the data slot to read a list of values from the database using an SQL query.
Line 10-23	Declares condition, rule and trigger logic that should be applied to the data returned by the SQL query. The logic slot fires and evaluates the logic and ends with a conclude statement of either *'true'* or *'false'*.
Line 25-26	Declares the action that should take place. In this example the action is to print to the screen the Alert_Text. The action slot then carries out the action if the logic slot concludes *'true'*. In this case, if the logic slot returns *'true'*, the Alert_Text is written to the screen.

There is considerable overlap between the Arden Syntax Medical Logic Modules (MLM) slot components and the novel ASK-agent components, which allow them to be transformed into the equivalent ASK-agent as shown in Table 4.3.

MLM Slot	**ASK-agent equivalent**
Evoke	The ASK-agent's action trigger to perform some task, or perform goal.
Data	Belief, the information the ASK-agent needs to achieve some objective.
Logic	Condition, precondition or trigger required for the selection of an appropriate plan.
Action	Execution of the plan.

Table 4.3 ASK-agent equivalent to MLM's slot components.

Table 4.3 ASK-agent equivalent to MLM's slot components explanation:

The '*evoke*' slot in the MLM is equivalent to the '*goal*' component in ASK-agents. It provides the commitment to achieve some objective. The '*data*' slot in the MLM is equivalent to an ASK-agent '*belief*'. It provides a framework for describing and classifying atomic object components in a domain. The '*logic*' slot in the MLM is equivalent to an ASK-agent '*trigger*', '*condition*', or '*precondition*' component. It provides the logic behind the selection of appropriate plans in order to achieve some goal. The '*action*' slot in the MLM is equivalent to the execution of the ASK-agents '*plan*' or '*sub-plan*' to achieve some objective or goal.

4.3 GLIF CIG components established as Agent components

To illustrate further similarities between ASK-agents and MLM approaches for handling guidelines, it is necessary to focus in more detail on GLIF. GLIF uses a hierarchical decomposition of guidelines in a series of Arden Syntax MLM workflow paths [Peleg_c et al., 2004]. In GLIF these MLM's are implemented using object-oriented software consisting of classes, attributes and the relationships among the classes. The GLIF system extends individual workflow paths developed in MLM's by networking their separate activities together. The original guideline authors did not specifically produce their guideline to be divided up into separate MLM module workflows, but this networking or linking of the separate MLM activities together permits GLIF to reproduce some of the intentions of the guideline authors [Peleg_c et al., 2004]. In Figure 4.1, top-level views of the main GLIF classes are presented. With the exception of the nested branch of the guideline model entity, all components relate to this decomposition of a single guideline workflow path into a CIG. The nested branch relates to where a guideline is linked or nested to another guideline using direct coupling.

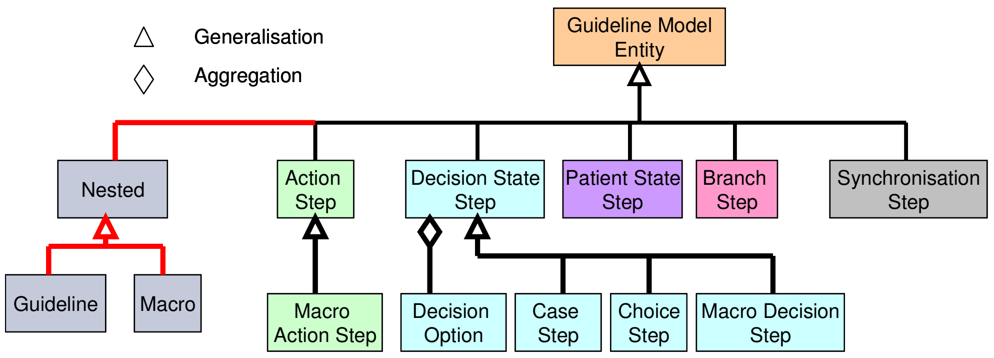


Figure 4.1 GLIF Computer-Interpretable-Guideline structure [Peleg_c et al., 2004]

Figure 4.1 GLIF Computer-Interpretable-Guideline structure explanation:
The CIG structure is the primitive skeleton framework behind GLIF's interpretation of a guideline. The main guideline entity can be decomposed into decision steps, action steps, branch steps and synchronisation steps.

The following subsections discuss these main GLIF components and provide customised ASK-agent equivalents for each. Additional components such as *levels of abstraction, ontology and language*, *description of services*, *levels of implementation* and *data type identification* are not directly linked to the operation of the CIG, but aid in their formation and are useful additional tools from a development perspective.

Decision step

A decision step represents a decision point or selection of an appropriate course of action in the guideline. The decision step can take many forms: conditional steps, branch steps, synchronisation steps.

A decision step using the ASK-agent approach is the selection of a plan, goal or partial plan. The condition or precondition slots are used to identify suitable plans. The ASK-agent inference logic then selects the most appropriate plans from the ones available with which to proceed using modal logic. The ASK-agent can use an internal event, message event or goal condition to trigger a plan. All of these can be formalised into similar operations as the GLIF decision step.

Action step

An action step represents an action to be performed in compliance with a guideline's workflow. Action steps contain tasks, and two distinct types of tasks can be modelled: medically orientated actions such as a recommendation for a particular course of treatment, and programming orientated actions such as retrieving data from an electronic healthcare record (EHR) or LIS using SQL.

An action step in the ASK-agent is an operation where a plan is executed. In some Java agent implementations a plan can be any Java program. It can be used to select patient information from a database using SQL or XML. Therefore, most medical procedures which can be converted into a Java application could be used.

Branching steps

Branch steps are used to allow multi-simultaneous paths of execution through the guideline take place [Peleg_d et al., 2003]. In some clinical and laboratory guidelines it is necessary to process two or more activities at the same time. This is accomplished by dividing the control thread.

Branch steps in the ASK-agent are possible using sub-goals and sub-plans. In some agent platforms a limited number of plans can be executed in parallel. In the other agent platforms plans must be executed sequentially. An agent can pause other activities until all branches have been executed.

Synchronisation steps

Synchronisation steps are branch steps where the multi-simultaneous paths must be performed at exactly the same time [Peleg_d et al., 2003]. Under certain conditions the separate control threads can be merged together at the end of the synchronisation step.

Synchronisation steps using the ASK-agent are possible by implementing sub-goals and sub-plans. Depending on the exact agent implementation used these can be executed in parallel and merged at the end of the step, or executed in series and rendezvous at the end of the step. While executing these steps all other agent activities are paused.

Levels of abstraction

'*Levels of abstraction*' forms part of the GLIF model where, after reviewing a guideline, the designer/modeller can develop a conceptual representation of the process (called Level A). When the designer is ready to refine the model a computable level of abstract (called Level B) is developed. At this point the GLIF model is not site specific. As soon as the model gets to the implementable level (called Level C) the application is deemed site specific. The first two levels (Level A & B) are developed using flow charts. This is similar to the way UML, as discussed in section 3.6, allows for software to be developed using graphical illustrations.

As discussed in section 3.6 a reported weakness of the agent platforms in general is the lack of agent specific modelling languages [Padgham et al., 2005; AUML, 2005]. Therefore, agent applications are still primarily constructed using raw Java code or XML text files. Agent Unified Modelling (AUML), MaSE, and Prometheus Modelling groups are actively working to address this issue.

Ontology and language

An ontology provides a structured format in which domain specific syntax and semantics are described. GLIF and all other CIG models only allow a single ontology to be developed for the entire system [Peleg_c et al., 2004]. All modules developed must use the same ontology in order to communicate with and understand each other.

Difficulties in communicating and sharing medical information between institutes, individuals or groups have generated a multitude of ontology and language implementations for example Galen [Rector, 2006], Tambis [Baker et al., 1999], UMLS [NLM, 2006], ONIONS [Gangemi, 1999], HL7 RIM [Dickinson, 2004; Beeler, 2001] and GENE [Gene, 2000]. These ontologies and language implementations specify various medical domains through an abstract conceptualised model of the real-world environment. This demonstrates that no unique "one-stop-shop" ontology for the medical domain exists, purely, because the ontology is based on an abstract conceptualisation model. The FIPA agent message structure recognises that in the real-world different ontologies exist, and instead of forcing a single ontology, it allows many to exist in the same environment and includes a framework to define and describe them. The FIPA ontology is composed of two parts, a vocabulary that describes the terminology of concepts used by agents in their realm of communication, and the classification of the relationships between these concepts, their semantics and structures [FIPA_c, 2003].

Levels of implementation

'*Levels of implementation*' relates to the hierarchy in which certain guidelines are positioned at different levels of patient care or diagnosis. A low level implementation activity could be for example, the guideline behind the validation of a single result analyte, say Alkaline Phosphates. A higher level implementation is where that result is combined with some other single result analytes, such as Bilirubin and GGT to aid in reporting a Liver Function Test. The Liver Function Test is then part of a higher level suite of tests for other medical disorder classifications. The GLIF model does not explicitly support '*levels of implementation*' but can implicitly support them through its unique method of identifying tasks.

Levels of implementation in the ASK-system are provided through the service description. '*ValidationType*' can indicate the type of service available while using the ASK-agent such as *for screening only* or *for comment.*

Description of services

Service descriptions are used in GLIF to provide a narrative searchable text slot that describes the service being provided by a MLM [Peleg_c et al., 2004]. If the service descriptions are adequately expressed, then other modules can search these slots in order to locate a service, for example *Liver Function Test*. This service description is in addition to the unique name given to the MLM and is similar to the trading service provided by CORBA [Vinoski, 1997].

Service Description is an integral aspect of the ASK-agent. The services are “lookup-able” using the DF (Directory Facilitator) and selection can be made under various categories.

Data type identification

The GLIF data type model allows for a variety of different data types to be used such as those shown in Table 4.4:

Primitive_Data_Type	=	Boolean, Int, Short etc.
Basic_Data_Type	=	A set of primitive types.
Literal Data Type	=	A data item of a fixed value
Data Item List	=	Run-time objects which allow referencing different data items in a single list.
Knowledge_Item	=	e.g. “Clinical_CHD” a specific clinical reference.
Variable Data Item	=	Represents data that needs to be instantiated at runtime: patient’s height, weight etc.

Table 4.4 GLIF CIG component data type identification [Peleg_c et al., 2004]

From an ASK-agent perspective, all of the GLIF data types can be accommodated easily in every one of the BDI agent systems discussed in section 3.4. In addition some agent platforms can implement any data type that can be represented by a Java Object.

4.4 The communication of medical and agent knowledge.

Chapter 3 explained that the underlying principle behind medical and agent knowledge communication is divided into three aspects; payload, envelope, and communication facility. It showed that the FIPA communication standard can accept any Java object, which could accommodate an XML, schema or object, as described in the CEN standard. It explained that the communication envelope from an agent’s standpoint has a similar principal aim to that of the medical envelope of accurate information delivery, but in addition must also provide a forum for the agents to interact at a social and operational level. Therefore, it is essential to verify if agent envelope data can support standard medical envelope data.

The FIPA message structure is made up of several parameters which must be present when agents pass messages. A list of the parameters used by the FIPA message standard is presented in Table 4.5, and each item in the index list is preceded by the prefix A. The list of parameters used by the medical communication standard, specifically ENV 13606-4:1999 Health informatics–Electronic healthcare record communication-Part 4 is given in Table 4.6. Each item in the index list is preceded by the prefix B. Both Table 4.5 and Table 4.6 provide crossover mapping between the standards in the right hand side column.

Index	**Parameter used in FIPA message**	**Description**	**Mapping to the ENV 13606-4:1999 Ref. Table 5.8**
A1	performative	Denotes the type of the communicative act of the ACL message. Inform, Request, Confirm	N/A
A2	sender	Denotes the identity of the sender of the message, that is, the name of the agent of the communicative act.	B3
A3	receiver	Denotes the identity of the intended recipients of the message.	B4
A4	reply-to	This parameter indicates that subsequent messages in this conversation thread are to be directed to the agent named in the reply-to parameter, instead of to the agent named in the sender parameter.	B5
A5	content	Denotes the content of the message; equivalently denotes the object of the action. The meaning of the content of any ACL message is intended to be interpreted by the receiver of the message. This is particularly relevant for instance when referring to referential expressions, whose interpretation may be different for the sender and the receiver.	Message Payload
A6	language	Denotes the language in which the content parameter is expressed.	B10
A7	encoding	Denotes the specific encoding of the content language expression.	N/A
A8	ontology	Denotes the ontology(s) used to give a meaning to the symbols in the content expression.	N/A
A9	protocol	Denotes the interaction protocol that the sending agent is employing with this ACL message.	N/A
A10	conversation-id	Introduces an expression (a conversation identifier) which is used to identify the ongoing sequence of communicative acts that together form a conversation.	B12
A11	reply-with	Introduces an expression that will be used by the responding agent to identify this message.	N/A
A12	in-reply-to	Denotes an expression that references an earlier action to which this message is a reply.	N/A
A13	reply-by	Denotes a time and/or date expression which indicates the latest time sending agent would like to receive a reply. **Notes:** The time will be expressed according to the sender's view of the time on the sender's platform.	N/A

Table 4.5 FIPA ACL Message Structure [FIPA_b, modified]

Index	Parameter used in ENV 13606-4:1999 message	Description	Mapping to the FIPA Ref. in Table 5.7
B1	identification of message by originator	Identifier assigned by the originator of a message.	A2 A10
B2	issue date and time of message	Date and time at which a message is issued by the sending application. Note: A mail service may store or batch messages and, as a result, the issue date and time of message need not be the same as the dispatch time to the intended recipient.	N/A
B3	EHCR source	A communicating party that sends EHCR information or receives a request for EHCR information.	A2
B4	EHCR destination	A communicating party that receives EHCR information or sends a request for EHCR information.	A3
B5	EHCR message related agent	A communicating party, other than the EHCR source or EHCR destination, referred to in a message to indicate their involvement in the process of communication.	A4
B6	urgency of message	Indication by sender of the urgency with which the receiver should deal with the request. Values: 0. Low 1. Normal 2. High 3. Immediate	N/A
B7	patient matching information	Information provided for the purpose of matching an EHCR message to a uniquely identified individual patient.	N/A
B8	message receipt acknowledgement request	Indication of whether or not the originator of a message requests an acknowledgement of receipt by the recipient. Values: 0. Never acknowledge. 1. Always acknowledge. 2. Acknowledge only if an error is detected.	N/A
B9	comment on message	A comment from the sender of the message.	N/A
B10	language	The language of the message represented by the abbreviations specified in ISO 639.	A6
B11	healthcare agents directory	A collection of information about any number of healthcare agents and/or healthcare agent in contexts that may be included in the message to enable subsequent reference without repetition of the detailed information.	N/A
B12	message reference	A reference from an EHCR message to a related message.	A10

Table 4.6 ENV 13606-4:1999 Message Structure [CEN_a, modified]

Analysis of the differences between the FIPA and ENV 13606-4:1999 standards shows six of the twelve medical transmission parameters have similar technical meanings with items in the FIPA standard. The main purpose of the FIPA ACL messaging structure is to allow agents to communicate effectively, and was not designed specifically for a medical application. However, FIPA specific implementations are free to include user-defined message parameters other than the items specified in Table 4.5. The semantics of these user-defined parameters is not defined by FIPA, and FIPA compliance does not require any particular interpretation of these parameters [FIPA_b, 2003]. The prefix "X-" must be used for the names of these non-FIPA additional items. Therefore, the FIPA messaging standard can be adapted with the addition of user-defined parameters, to provide a similar message model to that detailed in ENV 13606-4:1999, and yet retain its agent communication functionality.

To protect against future changes to the ASK-agent from a communication standpoint, it is important to also compare the FIPA standard to the new pre-standard CEN prEN13606-5 termed exchange models, which is Part 5 of the pre-standard documentation. This document is not in open circulation, but preliminary editions have shown a standard set of interfaces similar to that in the previous version of the ENV13606 standard. From the initial review of the documentation it is expected that an acceptable message payload will remain an XML document, schema, or object interface, and message envelope will be based on a number of mandatory and optional parameters in a similar way to the existing standard. As described above in the absence of an exact parameters list, the FIPA ACL message structure does allow for additional items to be added using the prefix "X-".

4.5 ASK-Agent supporting collaboration.

The three principal architectures used to provide the CIG hierarchical structure are rule base, direct coupling or blackboard systems. The rule-based systems are structured based on a nodal tree, where expert knowledge is combined with theoretical understanding and heuristic problem solving rules. Each branch of the tree starts with a decision or choice based on facts. Although firing rules based on presented facts during execution can be indicative of an illness, the rules which link them directly to a diagnosis do not reflect anything deeper than a casual understanding of human physiology. This is reflected in the fact that when changes are made to rule based systems, a considerable amount of effort is absorbed making the changes, but also discovering and rectifying rule firing complications in the system, which occur when rules are omitted or altered, changing the sequence of operation [Turban_a et al., 2005].

A traditional technique of combining a set of diverse software modules together is to connect them according to their data-flow requirements. The direct coupling architecture allows these modules to switch control to another module by a *direct call* or *link.* But the direct coupling model does not provide a clear representation of the overall problem. There is nothing more than relationship links used in order to solve the problem [Corkill, 2003] (See section 3.7.1). An alternative to direct coupling is the blackboard architecture (See section 3.7.2). The blackboard approach eliminates the communication issues raised by the directly coupled monolithic model, and gives a representation of the problem to be solved to all participants. But the blackboard does not have the capacity to indicate how group members can collaborate to solve a problem, and must select from partial solutions it already possesses. The expert knowledge sources (KS) do not have any awareness of other KS's, and are not able to communicate with any other expert KS's [Corkill, 2003]. They do not have, even at a fundamental level, any inner-understanding (introspection view) of other expert KS's (e.g., what parameters they use, what processes they perform, or what services they provide).

Agent architectures can be implemented in a similar fashion to the blackboard, where each agent acts using the BDI model implementation of the guideline. The overall view of the information about the problem is indicated on the shared area of the blackboard as shown in Figure 4.2 (Careful attention should be given to the inner boxes representing, control, blackboard, sub-plans and goals etc when viewing Figures 4.2, 4.3 and 4.4). This gives a representation of the problem to be solved to all participants, but it does not indicate how they can collaborate to solve a problem. The agents have no awareness of any other agent's capability, operations they perform, or information they require. The advantage of the model is that each agent executes in accordance with its BDI in parallel, acting on information presented in the shared area of the blackboard. As new information becomes available through the shared area, the agent immediately deliberates on it using its BDI, and returns a response to the blackboard without having to be prompted. However, this reliance on a centralised blackboard engine would be considered a step back, rather than a step forward in the eyes of devout agent theorists, and would surely meet with strong disapproval from same, as it hampers autonomous and independent behaviour of the agents.

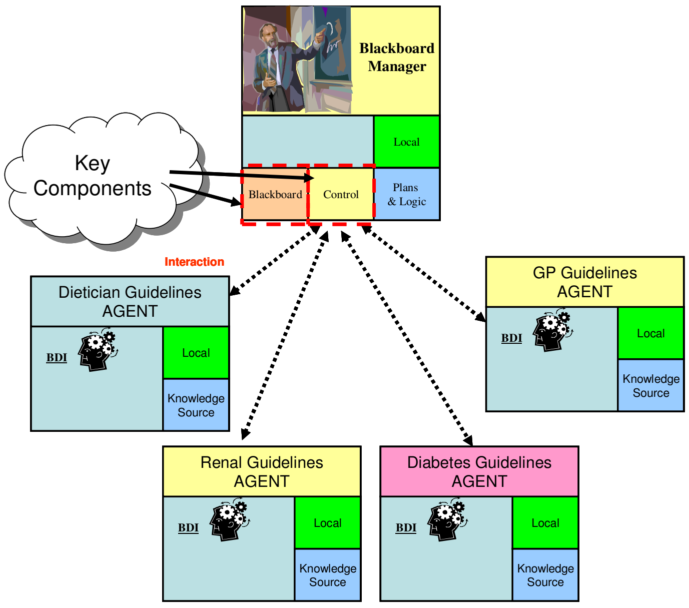


Figure 4.2 Agents implemented using the blackboard approach.

Figure 4.2 Agents implemented using the blackboard approach explanation:
Agents acting on behalf of guidelines, have local BDI execution where they access local data and knowledge sources. Each agent observes the blackboard central repository and uses that information to execute their BDI. The agent system can then communicate the outcomes of their action to the blackboard central repository for other agents to perceive. Agents act simultaneously by observing the blackboard, executing their BDI algorithm and reporting their outcomes for other agents to observe. However, the main controls of agent interactions are through the blackboard. Agents are not aware of any other agents in the system and cannot communicate with others directly.

The next incarnation of the agent involves the agents controlling the system execution, but retains the blackboard in order to share results with other agents as shown in Figure

4.3. The blackboard concept in this case does not have any logic, but is simply used as a repository for information, using a layered level of abstraction to handle the various types of ontology, languages or *groupage* of experts. This gives a representation of the problem to be solved to all participants, but it does not indicate how they can collaborate to solve a problem. In addition agents have no representation of capabilities of other agents, operations they perform, or information they require.

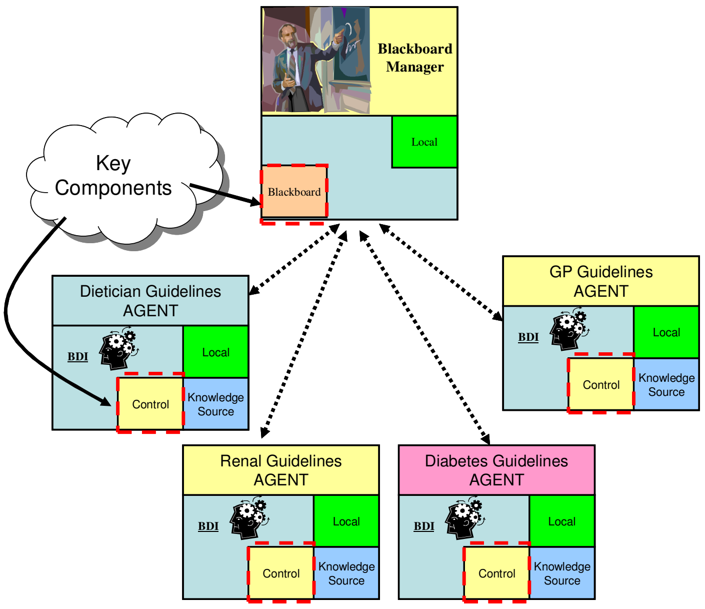


Figure 4.3 Agent with centralised blackboard.

Figure 4.3 Agent with centralised blackboard explanation:

All agents have the manifestation of a single guideline BDI encoded within them; therefore they act on behalf of the guideline they represent. The agent uses that BDI to provide the autonomous control over their actions, and have access to local data and local knowledge sources (e.g. plans, partial-solutions etc). Each agent observes the blackboard central repository, and uses information contained therein to execute their BDI. The agent system communicates the outcomes of their action to the blackboard central repository for other agents to perceive. Each agent is responsible for their own control and the blackboard is simply a central repository. The agents are not aware of each other and cannot communicate with other agents directly.

The final incarnation of the model proposed in this research, the ASK-agent, is based on the observation that if the blackboard is simply a repository of information, although layered to some degree, there is no absolute necessity for it to be in a single location. Working on this premise the blackboard system is replicated within each agent as shown in Figure 4.4. Using this novel ASK-agent approach, each agent's BDI implementation of the guidelines contains a list of locally derived beliefs, control logic (goals, plans, sub-goals, sub-plans and events) and a repository of layered information from other sources. The ASK-agents have direct communications between one another using the FIPA ACL messaging standard. This gives all ASK-agents a representation of the problem to be solved (i.e. through local blackboard, local beliefs, plans and goals), and indicates how the agents can collaborate to solve a problem. Every ASK-agent has an awareness of other agents in the group, and through introspection can find out information about their capability, the processes they perform and the information they require. This would be an application of the pull communication concept (see section 3.7.2). Using the agent FIPA communication standard the ASK-agents can correspond with one another in a social way in addition to the standard medical communication requirements.

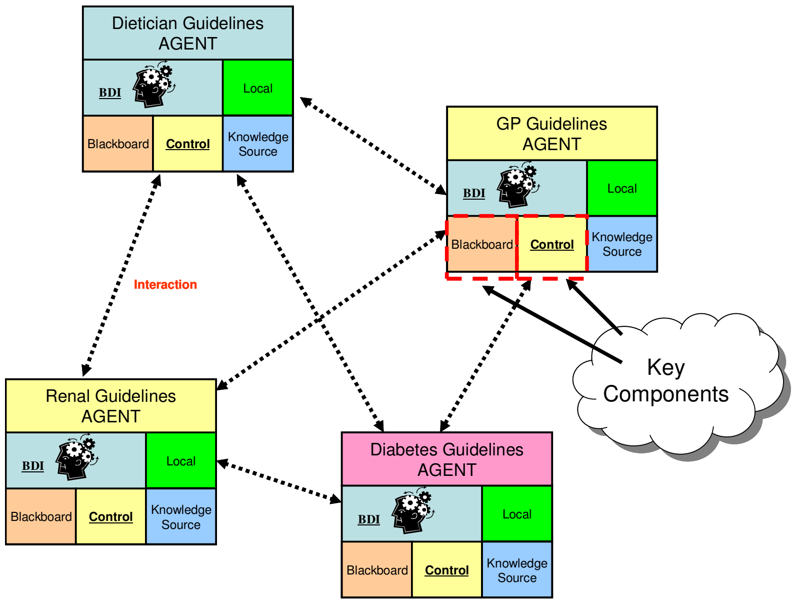


Figure 4.4 ASK- agent model.

Figure 4.4 ASK- agent model explanation:

The local blackboard for each ASK-agent is updated using message passing to ensure the replication of information. Every ASK-agent observes the local blackboard repository and uses that information to aid in their BDI execution. When an ASK-agent derives an outcome from their BDI execution, they can lookup the other ASK-agents in the group using the DF (lookup service similar to the *golden-pages*), and message pass the information directly to them. This would be an application of the push communication concept (see section 3.7.2). All ASK-agents are responsible for their own control and are aware of other agents in the group and can communicate directly with them.

The ASK-agent presented in Figure 4.3 allows for a problem representation to be formed by each agent. By fusing the link between the inter-agent problem representation stored in the agent blackboard and the intra-agent problem representation stored locally, a problem-solving dynamic begins to develop. The ASK-agents now have:

1. Awareness of other agents in the group.
2. A representation of what other agents are doing and deriving.
3. A method of interaction within the group using message passing.
4. Knowledge of other agent module capabilities through the DF.
5. The ability to search and locate services using the *golden-pages*.

The ASK-agent model is designed to act as a strong foundation for collaborating agent technology, and can provide the *similar domain knowledge* collaboration service (see section 3.7). However, the exact method of guideline collaboration and in particular the *overlapping domain knowledge* collaboration service still remains to be solved (see section 3.7). This is because the inference methods discussed in chapter 3 mainly focus on an individual's intelligence, and not necessarily on any collaborative intelligence. The concept of the agent paradigm is to synthesise elements found in human decision-making [Rao et al., 1995], therefore it is important to declare explicitly, not only how they act as an individual, but also how they collaborate and model this behaviour. Four key properties were suggested as fundamental requirements for any new agent based modelling. After comparing three agent-focused modelling languages (i.e. AUML, MaSE and Prometheus) it was shown, that the ability of the agents to socially interact was the least structured and weakest feature of the models. The three modelling languages performed social interaction in an ad-hoc fashion for each implementation [Padgham et al., 2005]. This makes it difficult to develop a system, such as the ASK-agent, which is strongly based on social interaction, using the software modelling languages. Even considering Corkill's challenges, such as representation, awareness, investigation, interaction, integration and coordination, which have been included within the ASK-agent model presented in Figure 4.3, it offers no structure that supports the activity of collaboration among guidelines.

4.6 Collaboration activity of agents

An agent provides rational inference through its BDI, which synthesises elements found in human decision-making, but the BDI does not explicitly detail how the agents can socialise, collaborate or form groups. Activity Theory (AT) focuses on the collaborative nature of separate autonomous systems such as individuals [Engestrom et al., 1999], on which agents are based, and have the capability to perform certain tasks. AT is an iterative process where an activity is developed from a simple to a higher level activity. Consider the example of a child learning to write. The child first begins with the activity of hand to eye coordination, where a pen is held correctly in the hand, and the child follows dotted lines on a page. As the child's coordination improves and becomes competent, the activity evolves to writing characters of the alphabet. With practice character writing begins to become automatic and evolves into writing words. The activity process evolves again to make words into sentences, and a further iteration from sentences to a paragraph and so on. The ability of the child to communicate with others through text changes at each level of activity.

Taking a medical example, a low level implementation activity could be the guideline behind the validation of a single analyte result, say Alkaline Phosphates. A higher level implementation is where the result is combined with some other analyte results, such as Bilirubin and GGT as part of a Liver Function Test. The Liver Function Test is then part of a higher level suite of tests for other medical disorder classifications. One guideline in itself does not cover the "*whole body*", but focuses on an abstract conceptualisation of the "*whole body*" from a specific viewpoint (e.g., disease, condition or organ). Another guideline relates to the same "*whole body*", but from a different viewpoint. Although the two guidelines are separate self-contained documents they are linked by virtue of their domain of discourse. Applying the AT structure, the first iteration of the medical domain of the liver blood tests can be illustrated as in Figure 4.5.

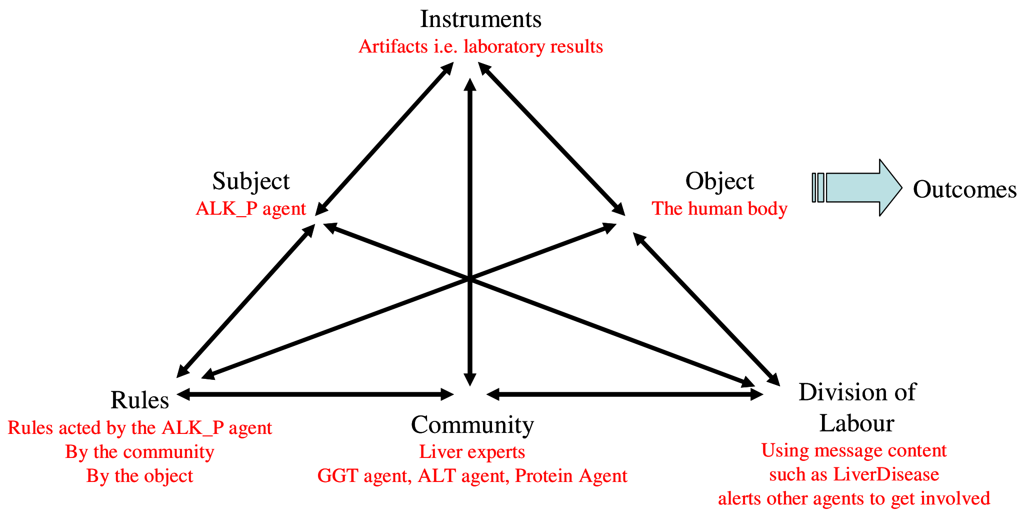

Instruments
Artifacts i.e. laboratory results
Subject
ALK_P agent
Object
The human body
Outcomes
Rules
Rules acted by the ALK_P agent
By the community
By the object
Community
Liver experts
GGT agent, ALT agent, Protein Agent
Division of Labour
Using message content such as LiverDisease alerts other agents to get involved

Figure 4.5 Structure of a Liver Blood Test activity system

Figure 4.5 Structure of a Liver Blood Test activity system explanation:

The subject component relates to the ASK-agent, in this case the "ALK_P agent (i.e. Alkaline Phosphates agent)". The community relates to the group of liver experts, in this case the GGT, ALT and Protein ASK-agents. The rules are divided up into rules contained in the ALK_P agent, which are performed by its BDI, rules imposed by the community and also rules imposed by the object namely the liver blood tests themselves. The division of labour is the content of message passing such as *LiverDisease* when alerted to other agents in the group. Some of these alerted agents may choose to note it and employ it, and others ignore it as it is not relevant to them. The instruments are the interface artefacts of laboratory results (i.e. ALK_P result).

The activity theory structure does not give an explicit software implementation, but acts as a design conduit for collaborative processes. The next iteration of the hierarchical activity is when the ALK_P ASK-agent forms part of a collaboration of other ASK-agents collectively known as liver function tests as illustrated in Figure 4.6.

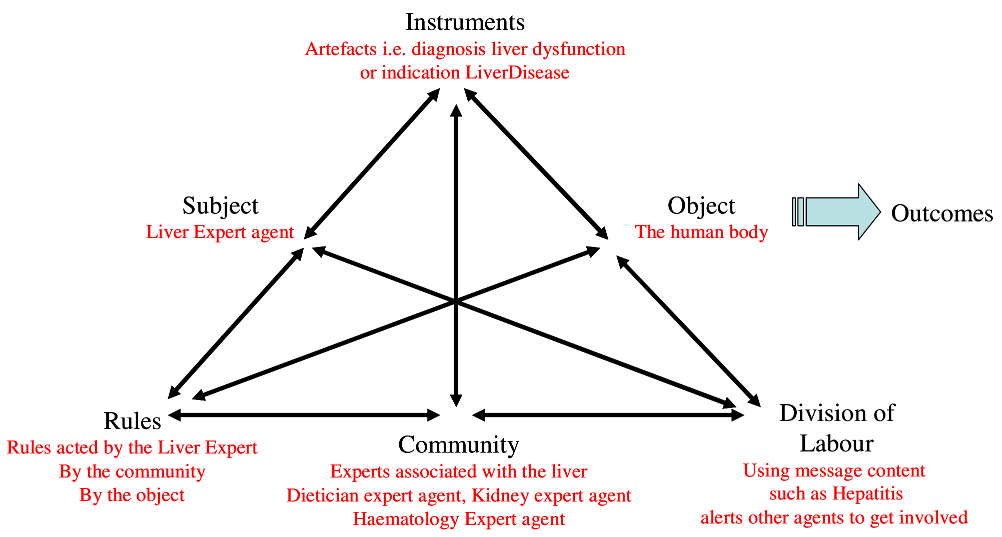

Instruments
Artefacts i.e. diagnosis liver dysfunction
or indication LiverDisease
Subject
Liver Expert agent
Object
The human body
Outcomes
Rules
Rules acted by the Liver Expert
By the community
By the object
Community
Experts associated with the liver
Dietician expert agent, Kidney expert agent
Haematology Expert agent
Division of
Labour
Using message content
such as Hepatitis
alerts other agents to get involved

Figure 4.6 Structure of the organ activity system

Figure 4.6 Structure of the organ activity system explanation:

The '*Subject'* refers to the ASK-agent in this case the *LiverExpert agent*. The '*community'* refers to the group experts associated with the liver, in this example Dietician expert, Kidney expert, and Haematology expert ASK-agents. The rules are divided up into rules contained in the '*subject*' *LiverExpert agent* to act on its BDI, rules imposed by the '*community*', and also rules imposed by the '*object*' namely the liver organ itself. The '*division of labour'* is the content of message passing such as *Hepatitis* when alerted to other agents in the group. The instruments are the artefacts of diagnosis for liver dysfunction or indication of Liver Disease.

4.7 A community of loosely coupled ASK-agents

The ASK-agent presented in Figure 4.6 supports agent collaboration and permits group problem-solving. The AT aspect allows the dynamic linking of information between layers by virtue of its respective viewpoints of the human body. But how would a community of agents know what is the problem they are trying to solve, and how would other agents in the community know when to coordinate, or collaborate, or assist other agents? It is not explicitly written into guideline that they are part of a larger collection of other guideline documents which all participate in describing and managing a large encompassing community (i.e. the human body, a community of organs). They are simply standalone autonomous documents specialising in a universe of discourse of a condition (not patient-centred), or disease or organ, which just happen to be involved in providing healthcare for a single patient. It is unlikely that a centralised medical planner could successfully coordinate all aspects (i.e. knowledge, reference ranges etc) of a human body (i.e. circulatory, muscular and respiration systems, and all the organs) and deliver appropriate healthcare for a specific patient. However, in spite of the practical difficulties associated with developing a centralised planning algorithm, the loosely coordinated efforts of the many clinical and laboratory guidelines, protocols and professionals solve the problem reasonably successfully.

As in all agent-based based systems, the underlying concept that binds the separate autonomous agent modules together in order to achieve some objective, is that the viewpoints of the domain the agents represent have some inherent overlapping of knowledge. The ASK-agent can provide a specialised service (i.e. condition specific guidance information) in a specialised domain (i.e. organs: heart, lungs, kidneys or diseases: diabetes or renal failure). But only if there is overlapping knowledge between agents, will they be able to collaborate on achieving a task. The solution to the global problem emerges from the collective activities of these independent and local agents.

The fundamental principal of the ASK-agent is to allow agents to collaborate without having to explicitly disclose their position as part of the large encompassing community, which would be a requirement if a centralised planner was used. To illustrate the concept consider the example of a jigsaw. A jigsaw piece has two discrete dimensions: the irregular shaped edge containing four sides, and the image printed on the face. To solve the puzzle it is possible to directly match the individual jigsaw pieces onto the whole jigsaw image as shown on the box. Each piece is then identified using the image on its face and placed in the appropriate position. This method requires a view of the whole system to be presented before starting, but involves no greater skill than straightforward pattern matching. An alternative approach is to use a combination of the localised image on the face of the piece and its four corners to match it to a suitable neighbour (i.e. matching the shape of the pieces together). Jigsaw assembly using these interfaces do not require the full picture to be known at the start. The only aspect that needs to be declared is the interface (i.e. communicate, coordinate and collaborate).

The jigsaw concept can be used as a metaphor to describe the ASK-agent system proposed in this thesis. Table 4.7 lists some of the main systems in the human body. Each of these systems has many guidelines developed about them specialising in certain aspects of associated conditions/disease or the organ/component itself, at different levels of abstraction. However, although these systems are separate they also interface and depend on each other in order to exist as illustrated in Figure 4.7. Examples of the interfaces are marrow in the bones of skeletal system which provide red blood cells in the blood, which support the absorption of oxygen by the blood, to feed muscle activity (e.g. heart). By using the jigsaw puzzle example and comparing it to the designed ASK-agent system, it shows that the ASK-agent was not intended to be a large all encompassing structure, but designed to be part of a very large collective group of small, social, independent focused expert knowledge (represented by each jigsaw piece). Each particular ASK-agent only needs to know its neighbours, which it can find and interact with using communications, in a similar way the jigsaw piece only needs to know another piece with similar edge profile that is compatible, not the whole picture. The agent does not need to know anything about any other sections of the larger jigsaw picture, only other agents it can directly communicate and interface with.

Item	**Description**
Circulatory system	The human heart is a component in the circulatory system, and made up of blood, cells, muscle etc.
Muscular system	The muscular system relates to the intercostals muscles, which act between the rib bones, expanding and contracting providing the mechanism for respiration, and heart muscles, which permit the heart to pulse to circulate blood around the body.
Skeletal system	The rib cage that encloses the lungs and heart is part of the skeletal system, and is made up of the sternum, ribs and clavicle bones, but also produces red blood cells.
Respiration system	The lungs form part of the respiration system, which includes the trachea, bronchi, bronchioles and alveoli, and permit the transfer of oxygen into and carbon dioxide out of the blood.

Table 4.7 Human body systems

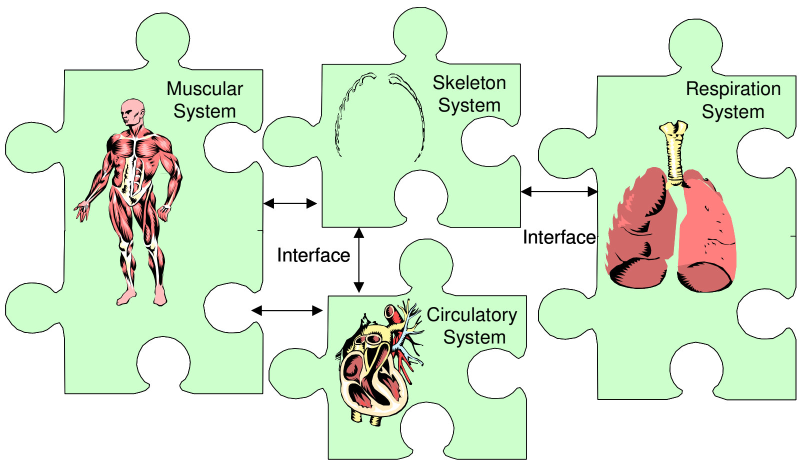


Figure 4.7 Jigsaw metaphor representing torso components

4.8 Functional requirements of ASK-agent

As discussed in the introduction (see section 1.1), the ASK-agent comprises of two components: (a) the base-level agent and, (b) the add-on customised features. This section focuses on the functionality requirements to be provided by the base-level agent platform. The base-level agent can be adapted or expanded to provide bespoke functionality for specific applications, in this case patient-centred validation services. The purpose of this section is to introduce and discuss the key functional features a base-level agent will require in order to support the development of the ASK-agent. These features are BDI execution, BDI structure contained in a single document, notation encoding, FIPA agent communication and reverse compatibility, as discussed in the following subsections.

BDI execution

The underlying concept of this research is based on a BDI agent's belief capturing the informational attributes, the desire capturing motivational attributes and the intention capturing the deliberative attributes of a guideline. The agency approach offers a fundamental synergy with the clinical and laboratory guidelines. An agent encoded with a similar BDI as the guideline will allow that agent to act autonomously on behalf of that guideline. Therefore, to aid in the demonstration of this synergy the chosen base-level agent must incorporate the BDI execution.

BDI structure contained in a single document

The agent must faithfully represent the goals, plans, events and ontologies inferred by a single guideline. When a guideline is updated or altered by its author over time, the agent must be able to be modified to reflect the changes. Agent systems where the BDI components are produced using separate document formats (i.e. where plans, partial solutions and beliefs are stored as separate entities) will make it difficult to ensure the single guideline is being faithfully represented after it has been updated over time. Therefore, the chosen base-level agent must incorporate one updatable document that can reflect the goals, plans, events and ontology changes.

Notation encoding

The encoding notation for information and logic contained within the BDI document should be easily interpreted and understood by the system's stakeholders. In the real-world environment the development of CIG workflow paths are accomplished mainly by clinicians and laboratory technologists and not computer technicians [Peleg_d et al., 2003], therefore, the chosen base-level agent shell must use an easily interpreted notation to encode guidelines.

FIPA standard agent communication

The comparison of the medical and FIPA communication standards (see section 4.4) demonstrated that the CEN standard can be accommodated using FIPA. It is therefore important that the chosen base-level agent employs the FIPA agent communications standard.

Reverse compatibility

Less advanced agent models change dramatically from one version release to another. The chosen base-level agent system should be at an advanced stage of development with commitments to provide reverse compatibility for new releases.

Table 4.8 summarises the agent shells discussed in section 3.4, and cross-references them to the functional requirement list detailed above. It focused on selecting agent shells that are at an advanced stage of development. The list of four agents included Jason (AgentSpeak), Jadex (JADE), 3APL and Procedural Reasoning Systems.

Functional requirements	**Jason & AgentSpeak**	**Jadex & JADE**	**3APL**	**Procedural Reasoning Systems (PRS)**
BDI Execution	√	√	√	√
BDI structure of the agent contained in a single document.	√	√	√	×
FIPA standard agent communications compliant	×	√	√	×
Notation	×	√	×	×
Reverse compatibility	√	√	√	√

Legend:	× - the requirement is not available or is not adequate.
	√ - the requirement is available or is adequate.

Table 4.8 Agent shells cross-references to the functional requirement list

Table 4.8 Agent shells cross-references to the functional requirement list explanation:
The PRS scored badly because it was developed using separate components to establish the BDI structure for one agent. These components were *act, function* and *database* files, which made it difficult to track changes, and considerably harder to configure than other agent shells. 3APL, Jadex and Jason are all advanced products incorporating the BDI structure, and they all use single documents to store beliefs and plans. However, 3APL and Jason both use a very explicit notation, which is not standardised outside the scope of their respective applications. Only 3APL and Jadex operate using the FIPA messaging standard, whereas Jason uses SACI which is based on KQML as discussed in section 3.4. The Jadex architecture is the only agent system that complies positively with all six aspects of the functional requirement list. Therefore, these arguments are the rationale for selecting Jadex as the agent architecture for the prototype proposed.

Additional aspects for using the Jadex platform are:

1. It uses XML to store belief, goals, plans, language and DF Description registration within a single user-friendly readable document.
2. Jadex is well published in peer review media.
3. Free for development.

4. Fully Java compliant (making it machine independent).
5. Provides a mobile format facilitating future expansion of the system (i.e. Lightweight Extensible Agent Platform (LEAP), [Claire, 2005]).

For the purposes of developing and designing the prototype in this thesis, all further ASK-agent descriptions will be developed and built upon from the Jadex agent platform.

4.9 ASK-agent components.

To provide the ASK-agent operation and functionality the base-level agent needs to be expanded. This is realised by providing three additional customised and compulsory components: a '*service description*', a '*set of compulsory beliefs*', and a '*set of compulsory actions*'. These three components extend the base-level agent and tailor its operation in order to manage the execution of individual ASK-agents representing guidelines and the management of inter-agent collaboration activity.

4.9.1 Service description

Service descriptions are used by some CIG models to identify modules by uniquely naming the service they provide and allow for service name searching, for example Liver Function Test.

The agent service description offers more facilities than those provided by the existing CIG models. The service description is an integral aspect of the ASK-agent approach and provides the ability to *lookup* services using the DF (Directory Facilitator), and the selection can be made under various categories all of which can be customised. During the creation of the ADF XML file a detailed service description is constructed similar to that shown in Code 4.1. The convention of service description properties has been developed to reflect the key selection criterion that clinicians use to choose an appropriate guideline to employ, or more specifically, which activities within that guideline could be utilised. This service description is written by the guideline encoder (the person converting the narrative guideline into the ASK-agent equivalent) and is used to publicise information about the ASK-agent in a '*golden-pages*' (i.e. the DF- Agent Directory Facilitator). Using the service description provided as part of FIPA an ASK-agent can publicise itself in the DF using its name, its type and its ownership by default. But additional customised properties can also be created. By structuring these properties a number of specific services can be explicitly declared to reflect the true operation of the ASK-agent as detailed below:

1. The service description property *name* is not the unique name of the ASK-agent (as that is provided by the AID[11]), but the name of a service the ASK-agent provides (e.g. SerumSampleValidation1 or SerumSampleValidation2 or UrineSampleValidation1 etc.). The FIPA communication standard allows each ASK-agent to have multiple service descriptions registered with the DF, but only one instance of each service description *name* is permitted in each agent.

2. The service description property *type* is used to identify the field of expertise to which the ASK-agent belongs (e.g. liver expert, kidney expert, and haematology expert). This allows ASK-agents to be part of an expert group, and receive messages from other agents that are relevant to the group. This is similar to a health board circulating notices to distinct expert groups, such as casualty nurses, consultants, GP or A&E doctors etc. This is an application of the push communication concept (see section 3.7.2), where an ASK-agent forwards information to others without being asked to do so. The information contained in the message is specifically relevant to that group of ASK-agents.

3. The service description property *ownership* is used to identify the guideline implementation owner. In most medical practice this is the name of the laboratory department who constructed the code, or where the service will mostly be used. This property provides another method of identifying groups similar to service description property *type*, but at an enterprise/departmental level.

4. The service description property *GuidelineReference* is a customised property that is used to identify the guideline name that the ASK-agent represents (e.g.

[11] *AID* is the Agent-Identifier which is a list of arguments used to uniquely identify an agent.

CholesterolGuideline, or ISBN, or the name of the group who developed it). This allows the ASK-agent to be traced and modified, when the guideline it is acting on behalf of, is modified or revised. The *GuidelineReference* component also allows for mutually exclusive guidelines to be eliminated from searches making the system more efficient.

5. The service description property *InformationNeeded* is a customised property that provides a list of the data the ASK-agent requires in order to process the validation request. Without this data the ASK-agent cannot perform the validation. Most generic clinical and laboratory guidelines allow for validation to be performed using different types and combinations of patient information. For example, validation using either patient's '*Gender and Height*', or '*Last Three Samples*' or '*Gender and Age*'. The *InformationNeeded* also presents the order in which the information should be sent (e.g. SerumALK_P;SerumALT;SerumGGT). The *InformationNeeded* titles can be developed to match the medical domain titles (e.g. *SNOMED*[12] codes), and/or those quoted in clinical and laboratory guidelines. This makes the guidelines more identifiable when being searched and suitable for a larger range of clinical settings. If some patient information is missing, alternative patient information, and perhaps an alternative validation method can be sourced and applied to the result (although in some cases perhaps not as accurately) (see section 3.8). This would permit the particular ASK-agent seeking information through the DF to know what information it needs to present and in what order it is to be presented in the message.

6. The service description property *ValidationType* is a customised property that defines the type of validation the ASK-agent can perform and it is possible for the same *ValidationType* property to be identified in the DF many times (e.g. SerumSampleValidation or UrineSampleValidation etc.) The *ValidationType* titles can be developed to match the service names commonly used in the

[12] *SNOMED* is a systematically organised collection of medical terminology codes covering most areas of clinical information such as diseases, findings, procedures, micro-organisms and pharmaceuticals.

medical domain. An ASK-agent can identify and locate many services that are provided within the guideline using this property. When searching for *ValidationType,* it is possible that the same agent can be identified many times within a single search, because it may use different methods and data in order to validate the sample.

7. The service description property *EndResultType* defines the type of end result a user of the service would expect to receive from the ASK-agent. Some guideline validation services can provide a comment in relation to the result, or in some cases provide a calculated numerical result (i.e. how damaged the liver is using a scale of 1-10). Another agent can query the *EndResultType* it would expect to receive from an ASK-agent searching the DF.

8. The service description property *ontology* defines the type of ontology the ASK-Agent uses to provide the service or collaborate with the service.

9. The service description property *protocol* defines the type of protocol the ASK-Agent needs to use in order to communicate with the service.

```
1   <servicedescription name="SerumSampleValidation1"
2           type="'LiverExpert"'
3           ownership="Microbiology_ImmunologyDept">
4           <property name="GuidelineReference">"GuidelineNo1a"</property>
5           <property name="ValidationType">"SerumSampleValidation"</property>
6           <property name="InformationNeeded">"SerumALK_P;SerumALT;SerumGGT"</property>
7           <ontology>ClinicalLaboratory</ontology>
8           <protocol> Snomed</protocol>
9   </servicedescription>
```

Code 4.1 Service description declared in the ADF XML file.

Code 4.1 Service description declared in the ADF XML file explanation:

Line 1	Declares the starting <servicedescriptions> XML tab for the ADF XML file.
Line 2-8	The service is given a *name*, *type*, *ownership*, *GuidelineReference*, *ValidationType*, *InformationNeeded*, *ontology* and *protocol*.
Line 9	Declares the <servicedescriptions> end XML tab for the ADF XML file.

4.9.2 Compulsory set of beliefs

The ASK-agent must implement in its beliefbase a set of mandatory beliefs to allow it interact with other group members. The minimum set of beliefs is shown in Code 4.2, and an explanation of each is detailed below:

1. The *'CurrentlyValidating'* belief is a Boolean flag which is set to *true* if the agent is busy and *false* if idle. Another agent can query ASK-agents by sending a message beginning with the text *'CurrentlyValidating'*. The ASK-agent receiving the message will immediately reply to the querying agent using a reply to sender message containing the contents of its *'CurrentlyValidating'* belief (see section 4.9.3).

2. The *'PlausibilityScore'* belief is used to store the plausibility index, expressed as a percentage, of the validation request which ranges from 0-100. A score of '0' would indicate the result is not plausible (i.e. 0% possibility the result being accurate for this patient), whereas a score of '100' would indicate the result is plausible (i.e. 100% plausible the result being accurate for this patient). As the ASK-agent progresses through the validation request, it alters the *'PlausibilityScore'* index to reflect its current belief as to the plausibility of the result under scrutiny.

3. The '*LiverDiseaseAgentResults*' beliefset is an example of a blackboard used to store all the messages received from other ASK-agents beginning with the text "*LiverExperts*". The name of the beliefset is not of importance, as an ASK-agent will have one or more of these blackboards allowing it to store information from various expert groups separately (e.g. KidneyDiseaseAgentResults or HeartDiseaseAgentResults etc). The beliefset stores the ASK-agent's unique identifier (AID) of the agent sending the information, and the actual message itself (both payload and envelope). This blackboard can be used to alter the ASK-agent's perception of its current validation activity, or can be used to revaluate the ASK-agent's outcomes against its peer agents in the group.

```
1    <beliefs>
2            <belief name="CurrentlyValidating"    class="boolean">
3                    <fact>false</fact>
4            </belief>
5            <belief name="PlausibilityScore"    class="int">
6                    <fact>-100</fact>
7            </belief>
8            <beliefset name="LiverDiseaseAgentResults" class="Tuple">
9                    <!-- <fact>new Tuple("AgentAID",LiverDisease;Comment")</fact> -->
10           </beliefset>
11   </beliefs>
```

Code 4.2 ASK-agent beliefs declared in the ADF XML file.

Code 4.2 ASK-agent beliefs declared in the ADF XML file explanation:

Line 1	Declares the starting <beliefs> XML tab for the ADF XML file.
Line 2-4	Establishes a belief named *"CurrentlyValidating"* of class '*Boolean*', which is used to indicate to other agents if this autonomous ASK-agent is currently active or idle. If *false* the ASK-agent is idle, if *true* the ASK-agent is active.
Line 5-7	Establishes a belief named *"PlausibilityScore"* of class *'int',* which is used to identify the plausibility score index of the validation request.
Line 8-10	Establishes a beliefset named *"LiverDiseaseAgentResults"* of class 'Tuple' which is the local blackboard used to post information retrieved from other agents in relation to the validation currently in progress.
Line 11	Declares the <beliefs> end XML tab for the ADF XML file.

4. Although most of the information communicated to the ASK agent is in the form of ACL messages from other agents, it is on occasion necessary for the ASK agent to access information directly from the LIS database such as mapping from SampleID to PatientID, or other predefined patient information, which is of importance to its execution. Therefore, it is essential that the ASK-agent has a method of locating this information quickly, but yet retain the context of the data. To this end, the user can define, *only at the time of encoding the ASK-agent,* an *archetype* query description which is stored in the Archetype Repository (AR) agent (see section 5.2), through which the ASK-

agent has access to exact patient data slots within the LIS Database. The ASK-agent requests that a specific archetype query is performed by the AR agent. The query data is returned to the requesting ASK-agent in the form of a message payload using accepted formats such as an XML document, schema, or an object interface. The implementation of an archetype is supported by the revised CEN pre-standard prEN13606-2 [CEN_e, 2007] and ensures the future proofing of the ASK-agent architecture.

4.9.3 Compulsory set of Actions

The ASK-agent must implement a number of mandatory actions to allow it to interact with other group members. The mandatory actions are automated responses to ASK-agent query messages from other agents and customised informative messages the ASK-agent wishes to send to a group of agents. Messages received from other ASK-agents would be considered an example of push communication (see section 3.7.2), where an agent forwards information to others without being asked to do so.

1. **Receiving messages from other agents.**
   When the ASK-agent is started, it runs an initial plan written in Java. The task of the plan is to check all incoming messages and search their content. If the message begins with the text *'CurrentlyValidating'* the ASK-agent must construct a reply to sender response containing facts stored in the *'CurrentlyValidating'* belief. An example illustrating this code is shown in Code 4.3. This style of filtering is implemented in a similar fashion for text messages containing '*PlausibilityScore*' where the ASK-agent must construct a return to sender response containing facts stored in the '*PlausibilityScore*' belief. This is an example of pull communication, where an ASK-agent is seeking specific information from another. The mandatory response is therefore an important aspect of the system and establishes a social fabric. The term '*reply to sender*' is where the reply is linked to the original message ID thread and is not a new message. This is done to ensure that sent messages continue with the same message ID thread and are traced at a network level.

If the ASK-agent is idle (i.e. *'CurrentlyValidating'* belief is *false*) and the receiving message begins with the text '*LabResult*' the ASK-agent must accept and process the result. The structure of the message payload is shown in expression (4.2)

LabResult;SampleID;Value (4.2)

Where the SampleID relates to the unique identification number of the sample assigned to it by the LIS, followed by the value of the result, separated by semi-colons. For clarity the type of laboratory sample can be determined from the sampleID and was therefore eliminated from this message.

The ASK-agent must implement one or more blackboards for storing messages from groups of other agents and it is mandatory that all of these informative messages are stored in these blackboards.

```
1       if (content.startsWith("CurrentlyValidating "))
2               {
3               StringBuffer buf = new StringBuffer();
4               buf.append(getBeliefbase().getBelief("CurrentlyValidating").getFact());
5               ACLMessage reply = msg.createReply();
6               reply.setPerformative(ACLMessage.REQUEST);
7               sendMessage(createMessageEvent(reply, buf.toString()));
8               }
```

Code 4.3 ASK-agent currently validating action using the plan.

Code 4.3 ASK-agent currently validating action using the plan explanation:

The code is an extract from a plan written in Java. After the agent receives a message, it converts it into a string and searches its content. If the message begins with the text "*CurrentlyValidating*" it executes the following:

Line 1	Is an IF statement which will be true if the message contents starts with the text "*CurrentlyValidating*" and the code between the brackets is executed.
Line 3	A string buffer is created called buf.

Line 4	The fact stored in the belief *"CurrentlyValidating"* is entered into the newly created string buffer.
Line 5	A FIPA standard *reply* message is constructed from the 'msg' which is the original message received by the ASK-agent.
Line 6	All FIPA messages must provide a performative component. The performative requirement adds a layer into the message system implementing the pull and push communications (see section 3.7.2). The standard ACL message performatives such as AGREE, CANCEL, CONFIRM, DISCONFIRM, FAILURE, INFORM etc. can be used.
Line 7	The message is sent back to the ASK-agent who sent the query.

2. **Sending medical content messages to other agents.**

When an ASK-agent communicates medical information with another ASK-agent it must be completed in a manner which complies with accepted medical information transfer protocols. This activity is handled by the FIPA compliant agent messaging network, and includes identification of message by originator, issue date and time of message, EHCR source, EHCR destination, EHCR message related agent, urgency of message, patient matching information, message receipt, acknowledgement request, comment on message, language, healthcare agents directory and message reference. These are all network *envelope* parameters that are handled by FIPA compliant agents. The payload of the message is the actual information the agent wishes to send to other experts using a specific language or ontology. In the pre-standard CEN prEN13606 Part 1 reference model, and Part 2 archetypes, an acceptable message payload can be an XML document, schema, or object interface. For the purposes of this implementation the structure of the message payload is simplified so its content can be displayed on the PC console, and is easy to understand. This proves the concept of the informative messages passing for demonstrating purposes without hiding information from the observer.

Messages sent from the ASK-agent will take the form as shown in expression (4.3) where all the components in the message are separated by semi-colons. Table 4.9 describes the meaning of each component.

"ExpertGroup;MedicalCondition;Probability;ResultType;Value;ResultTypeComment;
HowGenerated;ConfidenceLevel" (4.3).

Payload Message Component	**Description**
ExpertGroup	States group of experts to which the agent desires the information to be sent.
MedicalCondition	The technical description of the injuries, disabilities, disorders, diseases, syndromes, infections, symptoms and deviant behaviour that the message wishes the agents to be aware of.
Probability	States the probability of the medical condition being present. In this implementation the probability can be only one of three types: Possible — The agent does not have enough information to guarantee the medical condition is not present. Impossible — The agent has enough information to guarantee the presence of this particular medical condition is not possible in this sample. Probable — The agent has enough information that this particular medical condition is most likely present in this sample.
ResultType	States the type of result being commented on such as GGT, ALT or Alb.
Value	States the actual value of the result in a numerical form. The units of the result have been omitted for clarity, but would be dealt with using the payload ontology.
ResultTypeComment	States the agent's comments on the result. In this implementation the comment can be high, low or normal.
HowGenerated	States how the agent generated the belief the medical condition was present. Was it by calculation, or internal BDI. Calculation indicates a factual result and the internal BDI indicates it is the agent's opinion.
ConfidenceLevel	States the agent's confidence of the statement being true. The actual numerical confidence level value follows this component in the Payload Message separated by a semi-colon.

Table 4.9 ASK-agent message payload configuration

An example of the expression when viewed from the console is shown in expression (4.4) where the content of the respective components is detailed in Table 4.10.

"LiverExperts;LiverDisease;Possible;ALT;46;Calculated_value;High;Statement_Confidence;0.7" (4.4)

Payload Message Component	Content
ExpertGroup	LiverExperts
MedicalCondition	LiverDisease
Probability	Possible
ResultType	ALT
Value	46
ResultTypeComment	High
HowGenerated	Calculated_value
ConfidenceLevel	Statement_Confidence 0.7

Table 4.10 ASK-agent message payload equivalents using equation (4.4)

3. **Archetype query plan.**

   The ASK-agent must provide an archetype query plan (if required), which calls on the Archetype Repository agent to execute a predefined query of LIS data slots and return the context data back to the requesting ASK-agent in the form of a message (see section 3.5).

4.10 ASK-agent functionality

The purpose of this section and Appendix C is to expand the ASK-agent by carefully crafting vital functionality required by the proposed patient-centred validation approach. Clinical and laboratory guidelines are not presented by their authors in computer interpretable code using C++ or Java, but in narrative text, graphs and examples. Traditionally, CIG architectures provide a software layer that contains a selection of customised functions used to encode guideline information and knowledge into computer interpretable symbols [Peleg_d et al., 2003]. The customised functions represent standard operations used in guidelines such as branching, action, decision, scenario and sub-plans. These operations are used to translate the narrative text guideline into computer interpretable symbology, whilst trying to preserve their intended meaning. The person encoding the guideline develops the CIG workflow by choosing rules and functions from tailored menus in the CIG package. When the encoding is completed the CIG package then translates the user's application into code that can be executed on a particular software application such as C++ or Java. This '*CIG*' layered structure is shown in Figure 4.8.

The agent system on the other hand was developed for a broader audience, and not specifically for clinical and laboratory guidelines. The person encoding a guideline develops an application using the ASK-agent ADF XML file and Java plans. The user application is then translated by the agent platform and executed using Java as shown on the '*Agent Guideline System*' also in Figure 4.8. Appendix C carefully selects the bespoke functionality which is currently being employed by existing CIG methods, and reproduces and expands them for use in the ASK-agent, thus forming basic agent interpretable guideline building blocks.

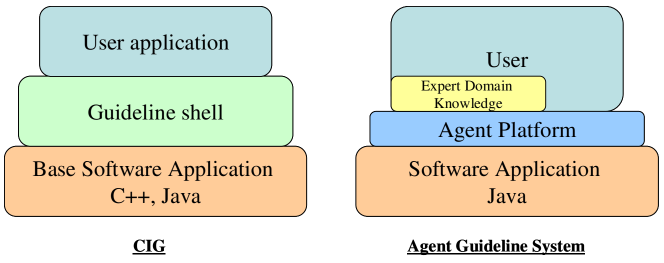


Figure 4.8 CIG and Agent Platform Layers

Figure 4.8 CIG and Agent Platform Layers explanation:

The CIG layers show the user's application is interpreted through a guideline shell that has special (customised) functions which can replicate guideline knowledge and information characteristics. This shell then converts the application into C++ or Java to allow it to be executed by the processor. The agent guideline system is a more general application that is being used in this thesis to replicate guideline characteristics and all the coding is performed by the encoder.

4.11 Discussion

The novel ASK-agent presented in this thesis is based on amalgamating two groups of functionality. The first set of functionality components are provided by a base-level, agent that focuses on the agents BDI, encoding, standard communication, installation, and reverse compatibility issues. The second functionality components are provided by expanding the underlying skeleton of the base-level agent to develop the ASK-agent that can act on behalf of clinical and laboratory guidelines. The purpose of this chapter was to examine generic concepts, functionality and methodology used by existing CIG models, and enhance and expand that functionality using ASK-agents.

Section 4.2 and 4.3 examined the capability of reproducing the guideline functionality using ASK-agents. Section 4.4 examined medical and agent data exchange functionality, and showed the FIPA communication is capable of being altered to allow communications to be completed in accordance with medical standards, and yet retain all of the FIPA agent's social communication fabric. Section 4.5 explored the possibility of altering the CIG dynamic from a centralised to distributed system by providing localised autonomous control and localised blackboard within the agent. Section 4.6 examined the activity of agent collaboration by comparing agent and human social activity and adapting the ASK-agent accordingly.

Using the concepts, functionality and methodology examined in sections 4.2-4.6 the base-level functional requirements list was produced in section 4.8. This list was used to compare the five BDI agent platforms introduced in Chapter 3 (see section 3.4). From that comparison the Jadex platform was chosen as the base-level agent. Section 4.9 developed the enhanced components that extended this agent platform. These components where motivation, knowledge and logic maps to guideline contents, and social fabric components for the ASK-agent's to collaborate and coordinate validation activities without the need for an all encompassing centralised plan. Section 4.9 also illustrated the mandatory components that have been incorporated into the ASK-agent in order to permit collaboration and social interaction, in addition to the guideline text to software conversion. Finally section 4.10 and Appendix C developed a suite of text to software conversion building blocks that can be used by developers who encode the ASK-agent, and which could be automated by future versions of the ASK-agent to provide customised menus.

Chapter

5

Patient-centred validation design

5.1 Introduction

The purpose of this chapter is to progress the architectures and methodologies analysed in chapter 4 by detailing the ASK-agent architecture and to illustrate an automated procedure to easily, consistently and efficiently convert clinical and laboratory guidelines into ASK-agent modules. Section 5.2 details the overall ASK-agent architecture and illustrates the interconnection between the various components. Section 5.3 describes the method of transforming a narrative guideline into an expert ASK-agent module. It shows how the Arden Syntax Medical Language Modules (MLM) are used and expanded in the ASK-agent to reproduce the domain specific perception contained in the guideline. This formalises the method of developing an ASK-agent module from any numerical narrative clinical and laboratory guideline. Expanding the ASK-agent encoding consistency theme, it is important to demonstrate that ASK-agents are capable of reproducing re-occurring principles, operations and methodologies used in existing guidelines. In 2003 a group of guideline encoding experts examined six CIG models, and published a paper to compare the models [Peleg_d et al., 2003]. This paper examined the principles, operations and methodologies used by six CIG modellers to encode guidelines into CIG's. Although the terminology and some special features varied between the models, Peleg et al showed that the main modus operandi of how they encoded guidelines into CIG's had some commonality. Using Peleg et al's approach Appendix C reproduces an equivalent of each principle, operation and method examined using an ASK-agent.

Finally section 5.4 presents various concluding remarks.

5.2 ASK-agent architecture

The architecture which captures the operation of the ASK-agent as discussed in chapter 4 is illustrated graphically in Figure 5.1. The message transport system (MTS) is based on the FIPA Agent Communication Language (ACL) standard with additional components being added to the message envelope parameters in order to support the medical communication standards. The agent management system (AMS) is a *mandatory component* of the base-level agent platform and is used to register and identify all agents within the platform. It is therefore, implicitly part of the ASK-agent architecture. It is a compulsory action for all agents (ASK-agents or otherwise) to register with the AMS. Only one AMS is permitted within each agent platform, but the AMS can be constructed using a federate network of sub-AMS applications. The AMS service is strictly a *white-pages* type facility, where only the unique agent-identifier (AID) and the local name of the agent are registered. There are no agent services descriptions or other attributes registered in the AMS. The directory facilitator (DF) on the other hand is a compulsory component of the ASK-agent architecture and all ASK-agents must register their available services with it. Each service is described using a set of different property attributes: *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Protocol* (see section 4.9). Only one DF is permitted within each agent platform, but the DF can be constructed using a federate network of sub-DF applications. The manager agent is the main *gateway* through which queued laboratory results in the LIS can be sent to the various ASK-agents for validation. Its main purpose is to accept validation queries from the LIS, search for a number of ASK-agents to begin the validation service, and report the results back to the LIS when the agents have completed the validation. One or more managers can exist on the system simultaneously, but the quantity depends on the processing and network resources available to the user. The archetype repository (AR) agent is the location where each ASK-agent's archetype queries are stored as separate templates. The archetype query allows the encoder to create a specific ASK-agent data structure definition which provides intelligent querying of LIS data. The ASK-agent implements the capabilities, beliefs, goals, plans, events, languages, ontologies and services descriptions as described in section 4.9 and acts as a computerised version of a guideline. Only one ASK-agent should exist for each particular guideline, but many instances of the same ASK-agent can exist simultaneously, with the upper limit on ASK-agents depending on the processing and network resources available to the user.

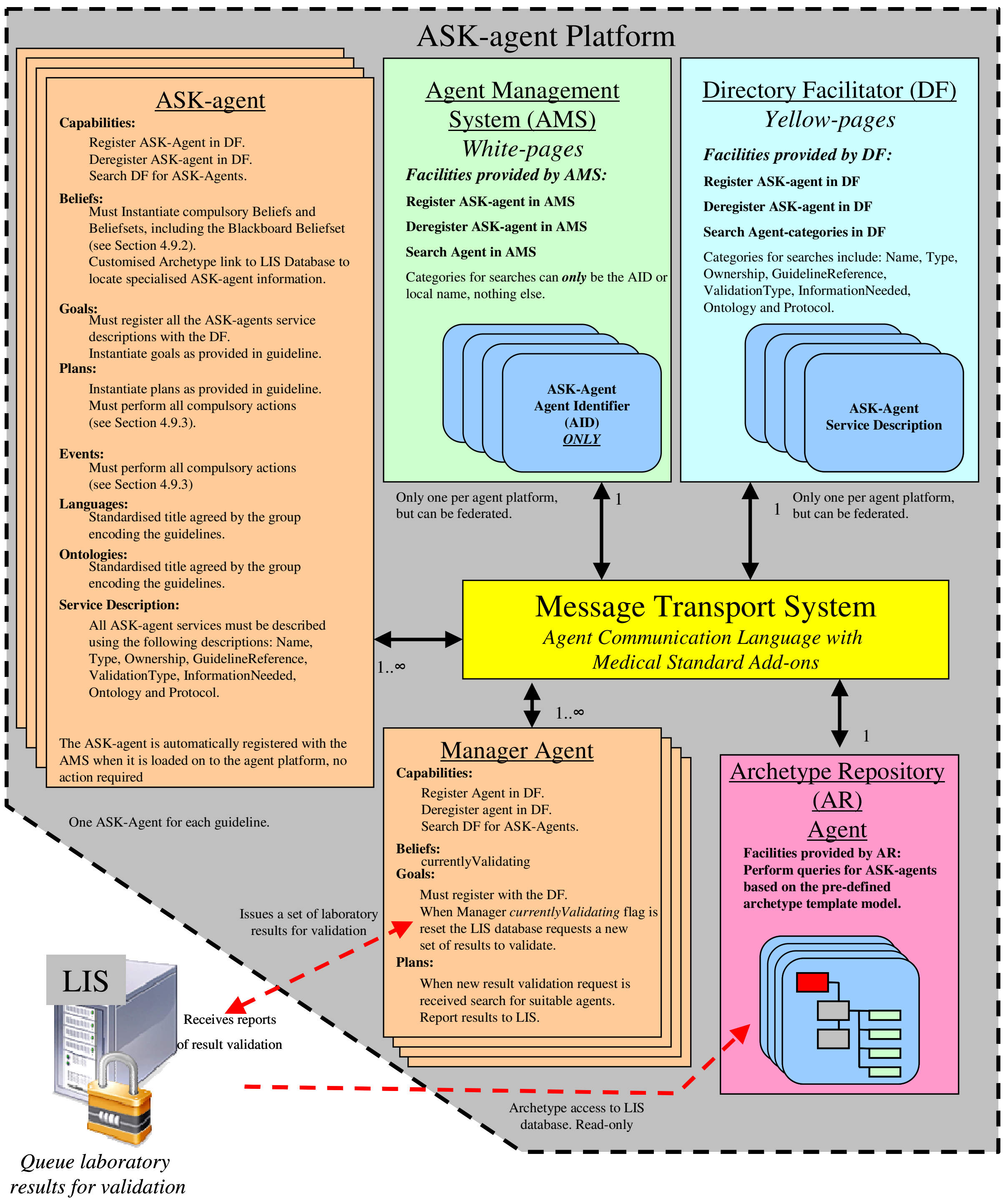

ASK-agent Platform
ASK-agent
Capabilities:
Register ASK-Agent in DF.
Deregister ASK-agent in DF.
Search DF for ASK-Agents.
Beliefs:
Must Instantiate compulsory Beliefs and Beliefsets, including the Blackboard Beliefset (see Section 4.9.2).
Customised Archetype link to LIS Database to locate specialised ASK-agent information.
Goals:
Must register all the ASK-agents service descriptions with the DF.
Instantiate goals as provided in guideline.
Plans:
Instantiate plans as provided in guideline.
Must perform all compulsory actions (see Section 4.9.3).
Events:
Must perform all compulsory actions (see Section 4.9.3)
Languages:
Standardised title agreed by the group encoding the guidelines.
Ontologies:
Standardised title agreed by the group encoding the guidelines.
Service Description:
All ASK-agent services must be described using the following descriptions: Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology and Protocol.
The ASK-agent is automatically registered with the AMS when it is loaded on to the agent platform, no action required
One ASK-Agent for each guideline.
Agent Management System (AMS)
White-pages
Facilities provided by AMS:
Register ASK-agent in AMS
Deregister ASK-agent in AMS
Search Agent in AMS
Categories for searches can only be the AID or local name, nothing else.
ASK-Agent Agent Identifier (AID) ONLY
Only one per agent platform, but can be federated.
1
Directory Facilitator (DF)
Yellow-pages
Facilities provided by DF:
Register ASK-agent in DF
Deregister ASK-agent in DF
Search Agent-categories in DF
Categories for searches include: Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology and Protocol.
ASK-Agent Service Description
1
Only one per agent platform, but can be federated.
Message Transport System
Agent Communication Language with Medical Standard Add-ons
1..∞
1..∞
1
Manager Agent
Capabilities:
Register Agent in DF.
Deregister agent in DF.
Search DF for ASK-Agents.
Beliefs:
currentlyValidating
Goals:
Must register with the DF.
When Manager currentlyValidating flag is reset the LIS database requests a new set of results to validate.
Plans:
When new result validation request is received search for suitable agents.
Report results to LIS.
Archetype Repository (AR) Agent
Facilities provided by AR:
Perform queries for ASK-agents based on the pre-defined archetype template model.
Issues a set of laboratory results for validation
Receives reports of result validation
LIS
Queue laboratory results for validation
Archetype access to LIS database. Read-only

Figure 5.1 ASK-agent architecture

5.3 *Procedure to implement an ASK-agent.*

The transformation procedure to implement an ASK-agent module from a specific narrative guideline is based on the cycle shown in Figure 5.2. This procedure expands on the Medical Language Modules (MLM) as discussed in section 2.6 to provide a map of MLM features into the Agent ADF XML file and Java plans. Each component is discussed in the following subsections.

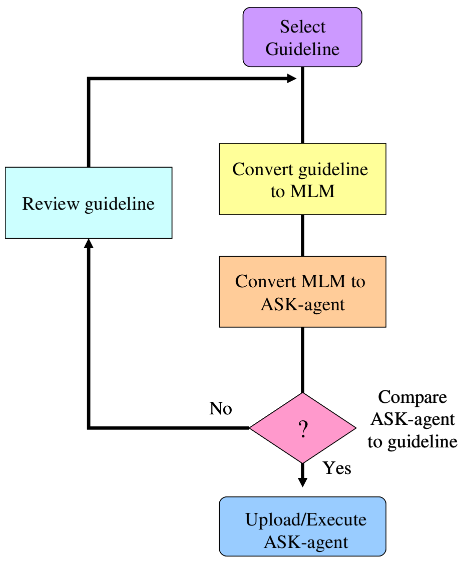
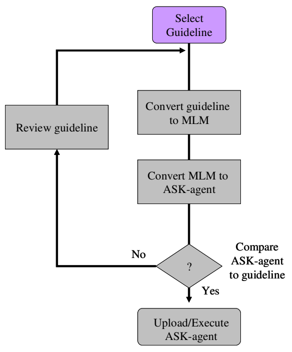


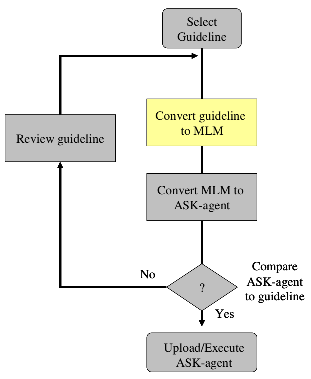
Figure 5.2 Encoding of expert agent procedure

5.3.1 *Select guideline*



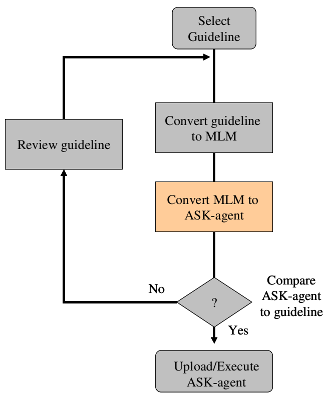
The guideline must contain overlapping knowledge with other guidelines and medical disciplines which use the laboratory facilities. As is generally the case with CIG systems, the guidelines content for ASK-agents should be narrative and numerical based rather than dependent on graphs and/or photographical images in order to describe its knowledge and logic.

5.3.2 *Convert guideline to MLM.*



The Arden Syntax Medical Logic Module (MLM) rules are applied to the guideline where it is divided up into separate workflow paths using the '*evoke*', '*data*', '*logic*' and '*action*' slots. Each guideline workflow path is represented by a single MLM. The linking, merging and crossover for different MLM modules in the domain space is not completed in this phase. Examples of this phase were illustrated in section 4.2, and Table 4.2 provided an equivalent Arden Syntax Medical Logic Module (MLM).

5.3.3 *Convert MLM into ASK-agent.*



The ASK-agent expands the MLM by using a direct mapping of MLM components to ASK-agent components as shown in Table 5.1. Code 5.1 illustrates how the ASK-agent ADF XML file accommodates the MLM slot components. The standard MLM structure does not have any component relating to the motivational stance of the MLM module's workflow path. In the existing CIG architectures this is completed centrally in the CIG engine and not held as a distributed component of the individual MLM modules. This motivation and desire is extracted from the original narrative guideline document where the ASK-agent components of achieve, query and maintain goals, and the selection of plans and sub-plans using meta-level and modal reasoning are added to the BDI. This provides the driving force behind the ASK-agent's activity. The ASK-agent also uses other aspects that are not explicitly declared in the guidelines, such as language, ontology and service descriptions.

MLM Slot	**ASK-Agent Component**
Evoke	The ASK-agent's action trigger to perform some task, or perform goal.
Data	Belief, the information the agent needs to achieve some objective.
Logic	Condition, precondition or trigger required for the selection of an appropriate plan.
Action	Execution of the plan.
Not available	*NEW* Achieve Goal – motivates the ASK-agent to achieve a specific goal to reach some desired state, such as determine patient gender, age, PatientID. The use of the <targetcondition> and <failurecondition> specifies cases in which a goal can be considered as achieved, or failed, and an alternative course of action can take place.
Not available	*NEW* Maintain Goal – motivates the ASK-agent to maintain a specific condition (e.g. maintain the plausibility value above 60%)
Not available	*NEW* Query Goals – motivates the ASK-agent to seek alternative avenues on an IF basis (e.g. test alternative paths before committing to the path, such as would knowing the gender of the patient alter the outcome?)
Not available	*NEW* Meta-level reasoning – if more than one path is possible the ASK-agent must choose the most appropriate course of action to take. This could include performing query goals to obtain some supportive information in order to aid the decision (e.g. patient has liver disease, before course of action could be selected, which one should be investigated first)
Not available	*NEW* Modal reasoning – if data stored in the beliefs or received in a message has a level of truth, but cannot be established as 100% true or false, then the ASK-agent can weight its selection of an appropriate action (e.g. the patient could have kidney failure, probability of 60%, but the patient could also have liver failure, probability of 55%)
Not available	*NEW* Language – the guideline targets a specific audience who use a particular phraseology (e.g. the specific meaning of high blood pressure in relation to this specific audience). The ASK-agent allows for this to be defined in a bespoke library. This language definition can be used by many agents.
Not available	*NEW* Ontology – The construction of sentences used for transmitting messages and processing of information by the ASK-agent can be defined for a specific audience. This allows agents to transmit messages to specific types of ASK-agents by choosing a particular ontology when sending a message.
Not available	*NEW* Service description – the ASK-agent permits the defining of a list of services, which can be accomplished by the information contained therein. These services can be catalogued and looked-up using *golden-pages* type searches.

Table 5.1 MLM component map to agent components

```
1    <!--Description of Agent First Heading -->
2
3    <!--      <H3>Description of Agent Second Heading</H3>
4              <H4>Description of Agent Third Heading</H4>
5    -->
6    <agent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
7              xsi:noNamespaceSchemaLocation="http://jadex.sourceforge.net/jadex.xsd"
8              name="NameOfAgent"
9              package="">
10
11   <imports></imports>
12
13   <capabilities> </capabilities>
14
15   <beliefs> Data MLM  Slot </beliefs>
16
17   <goals> Evoke MLM  Slot, Achieve Goal, Maintain Goal and Query Goals </goals>
18   precondition, conditions and triggers activating the goal are represented by the Logic MLM  Slot
19   <plans> Action MLM  Slot </plans>
20   precondition, conditions and triggers activating the plans are represented by the Logic MLM  Slot and
21   Evoke MLM  Slot and Meta-level reasoning,
22
23   <events> </events>
24
25   <languages> Language </languages>
26
27   <ontologies> Ontology </ontologies>
28
29   <expression> </expression>
30
31   <servicedescription> Service description </servicedescription>
32
33   <properties> </properties>
34   </agent>
```

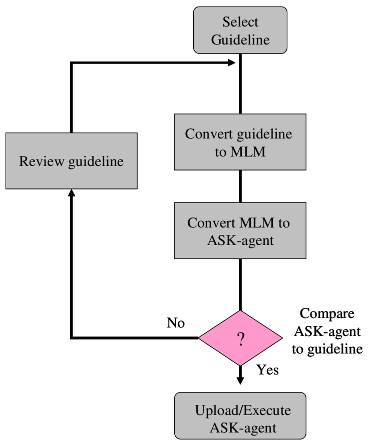
Code 5.1 MLM component map to ASK-agent components

Code 5.1 MLM component map to ASK-agent components explanation:

Line 1-5	Describes the headings used for identifying the ASK-agent modules. It is free text and is similar to comments used in computer applications.
Line 6-9	Describes agent naming and package declaration elements of the ASK-agent system.
Line 11	Identifies the imports in the ADF XML file.
Line 13	Identifies the capabilities in the ADF XML file.
Line 15	Identifies the beliefs section of the ADF XML file that is equivalent to the Data MLM Slot.

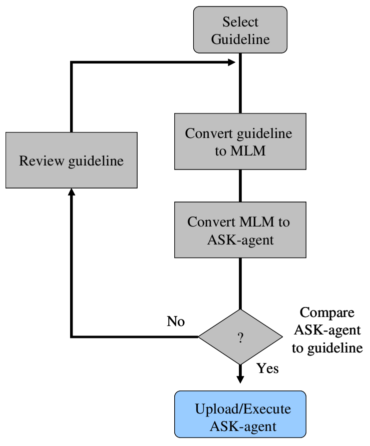
Line 17	Identifies the goals section of the ADF XML file that is equivalent to the Evoke MLM Slot.
Line 18	The triggering of a goal is achieved via the precondition, conditions and triggers activating the goal that are represented by the Logic MLM Slot.
Line 19	Identifies the plans section of the ADF XML file that is equivalent to the Action MLM Slot.
Line 20	The triggering of a plan is achieved via the precondition, conditions and triggers activating the goal that are represented by the Logic MLM Slot.
Line 23	Identifies the events section of the ADF XML file. This component is not matched in the MLM structure, but can be used to aid in development of the agent's autonomous nature.
Line 25	Identifies the languages section of the ADF XML file. This component is not explicitly declared in the MLM structure, but is used to define objects in the chosen domain.
Line 27	Identifies the ontology section of the ADF XML file. This component is not explicitly declared in the MLM structure, but is used to define syntax and data structures within the guideline.
Line 29	Identifies the expression section of the ADF XML file. This component is not explicitly declared in the MLM structure, but aids in the execution of logic. The expression component allows for data searches.
Line 31	Identifies the servicedescription section of the ADF XML file. This component is not explicitly declared in the MLM structure other than in the text comments about the service the MLM provides. In this system the creative structure of the service descriptions allows the accurate and context rich searching of suitable agents.
Line 33	Identifies the properties section of the ADF XML file. This is an internal component of the agent system allowing it create warning messages to the computer console for debugging or error identification.

5.3.4 Compare ASK-agents to guideline



After the ASK-agent ADF XML file and associated plans are developed and tested, the implementation needs to be examined to ensure it represents the intention, logic and knowledge of the original guideline. In the same way guidelines are autonomous self-contained entities, the ASK-agent must act in a similar fashion. The ASK-agent knowledge and logic can only be accessed through messages. The testing of the ASK-agent is completed by switching the logging level to fine within the properties section, so all ASK-agent processing is displayed at the operator's console. Then the testing is divided into two phases. Phase 1: the human designer sends synthetic patient data messages to the ASK-agent using message passing in isolation, and analyses the ASK-agent's response. Phase 2: the ASK-agent is loaded into a replica of the existing ASK-agent platform, where other ASK-agents are also present. Additional synthetic patient data is then applied to the group, and console output is analysed. If the ASK-agent implementation of the guideline achieves the correct responses, the logging level is switched off and the ASK-agent is then uploaded into the main agent platform and is ready for use.

5.3.5 Upload and execute the ASK-agent



The developed ADF XML file and Java plans that represent the guideline are loaded into the Remote Agent Management. Using the Remote Agent Management GUI each agent is selected in turn using the "Start New Agent" button as shown in Figure 5.3 and loaded into the manager. Once installed the agent is able to act as part of the social group. The uploaded agents can be switched on, off or suspended by using the JADE Remote Agent Management GUI.

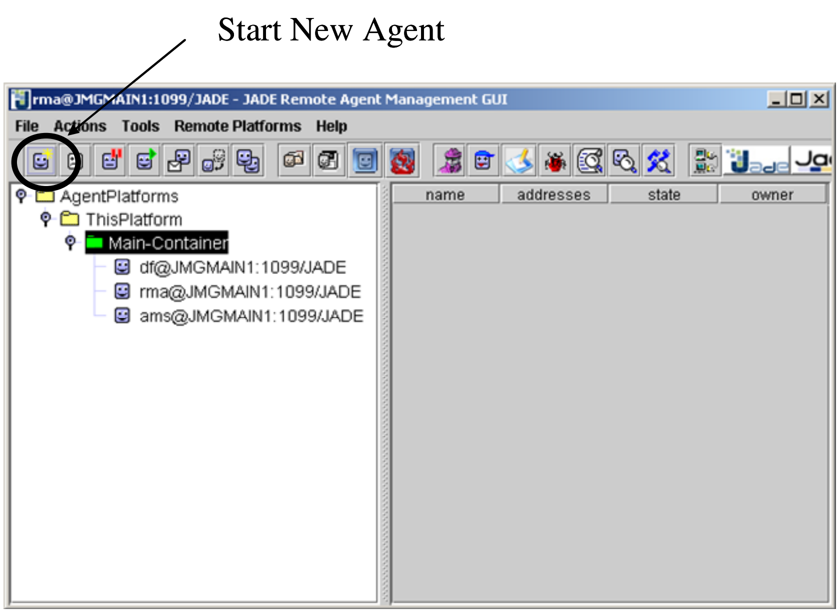
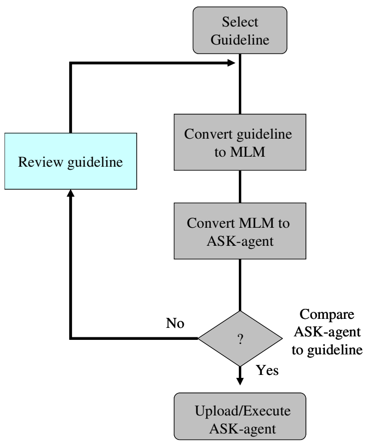


Figure 5.3 Remote Agent Management GUI.

5.3.6 Review guideline

If the autonomous action of the agent module does not represent the intended operation and understanding of the original narrative guideline the agent encoding should be reviewed. From the proof of concept implementation experience, the majority of errors relate to the encoding of the motivational stance using goals and adjusting of the meta-level reasoning. This manual review is repeated until the agent represents the true operation of the guideline.



5.4 Conclusion

This chapter presents the underlying components used as part of the ASK-agent architecture. The ASK-agent template component is used to describe inter/intra components of clinical and laboratory guidelines. This component can be reproduced many times within the architecture to represent the various guidelines used. The AMS component registers the agents using their local name and the AID, and only provides a *white-pages* look up service. Comprehensive ASK-agent searching is provided through the novel service description properties which are registered with the DF component. The

DF acts as a search repository for agent service descriptions using the attributes: *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Protocol*. These description properties reproduce the selection criterion used by clinicians and laboratory technologists when selecting appropriate guidelines and activities which can be applied to known patient data. The archetype repository agent component is used as a conduit through which ASK-agents can request the execution of pre-defined intelligent searches of the LIS in order to retrieve patient specific data in a context preserved way, which is returned to the ASK-agent in the form of a message. The manager agent acts as the gateway between the LIS and the ASK-agent platform. Upon receiving a result which was queued by the LIS, the manager searches for, and informs, available ASK-agents of the information it has on the patient and the validation begins. When the ASK-agents have concluded their deliberations on the presented information they produce a report which is returned back to the LIS for storage and distribution to the party who requested the information.

The accurate capturing of the information and logic contained with guidelines is of critical importance to the operation of the system and is completed using iterative steps. To this end the thesis continues by describing the procedures a user would complete in order to promptly, consistently, and efficiently convert clinical and laboratory guidelines into ASK-agent modules. These steps were crafted to extract the different levels of information and logic contained within the clinical and laboratory guidelines and map them into the fundamental ASK-agent.

Chapter

6

Patient-centred validation service implementation

6.1 Introduction

The purpose of this chapter is to demonstrate the operation of the ASK-agent architecture through a working model of the system when applied to a real medical problem. The model is based on validating Liver Function Test (LFT) results which are used by clinicians as a key indicator to the source of anaemia for chronic diseases. Therefore, it is important to describe the medical domain in more detail so execution nuances of the ASK-agent implementation can be clearly identified.

Marrow of large bones in the human body produces Red Blood Cells (RBC) (containing haemoglobin) at a rate of about 2 million per second, which is distributed throughout the body via the blood [Kumar, 2002]. The purpose of RBCs in the blood is to transfer oxygen to the tissues and cells, and removal of carbon dioxide (a waste product) from cells and transport it to the lungs to be exhaled. Since all human cells depend on oxygen combined with the removal of carbon dioxide for survival, if there are insufficient RBCs present in the blood the body begins to malfunction. Anaemia is the most common disorder of the blood and is described as a deficiency of RBC and/or haemoglobin [NAAC, 2005; Kumar, 2002]. Varying degrees of anaemia exist which are caused by different types of acute or chronic disease/illness. The presence of anaemia on its own does not correlate to a particular underlying disease/illness and in some cases may be normal as dictated by the patient's metabolism.

When a clinician identifies a patient with anaemia, they try to identify the cause. It is not the function of the laboratory to determine the cause, but they can provide information to aid the clinician in making a diagnosis or manage a patient's healthcare. The three main classes of anaemia include excessive blood loss (acutely such as a haemorrhage, or chronically through low-volume loss), excessive blood cell destruction (due to a virus or chemical imbalance), or deficient red blood cell production. Excessive blood loss can be determined if there was blood in the urine, stool, or a visible presence of blood outside the body (e.g. bandage). Internal haemorrhage can also be detected by the clinician through touch, visible bruising, x-ray or ultrasound. However, if none of these signs are present then the clinician needs additional evidence to determine the cause of the anaemia. It would also be standard practice for the clinician to reaffirm their diagnosis by eliminating the possibility of the other form being present too. Both excessive blood cell destruction and deficient red blood cell production forms of chronic anaemia in particular can be caused by a variety of different illnesses and diseases such as kidney disease, liver disease, alcohol disease and inflammatory disease as discussed below:

Anaemia of chronic kidney disease

Healthy kidneys act as blood filters and remove waste from the blood, which the patient disposes of through the passing of urine. Most serum proteins are too big to pass through the kidneys' filters into the urine unless the kidneys are damaged. The kidneys also control blood pressure and to a lesser extent produce RBCs. When the kidneys are damaged, they are unable to remove waste from the blood or make RBC's, or this action is compromised to some degree, and this condition is termed *Anaemia of chronic kidney disease.*

Anaemia of chronic liver disease

RBC's generated in the bone marrow are destroyed in the spleen when they get old or damaged. This releases hemoglobin, which is broken down to heme, as the globin parts are turned into amino acids. The heme is then turned into unconjugated bilirubin in the spleen and attached to albumin and sent to the liver, before being excreted into bile ducts. Patients usually have jaundice if there is a breakdown in a component of the bilirubin metabolism and the patient may have a problem of liver or RBC production and destruction. This condition is termed *Anaemia of chronic liver disease.*

Anaemia of chronic alcohol disease

Alcohol affects many organ systems of the body, but perhaps most notably affected are the central nervous system and the liver. Almost all ingested alcohol is metabolised in the liver and excessive alcohol use can lead to acute and chronic liver disease. Liver cirrhosis resulting from alcohol abuse is a leading cause of death world wide. Cirrhosis is a consequence of chronic liver disease characterised by replacement of liver tissue by fibrosis scar tissue as well as regenerative nodules, leading to progressive loss of liver function. Cirrhosis is most commonly caused by alcoholism and hepatitis C, but has many other possible causes. This condition is termed *Anaemia of chronic alcohol disease.*

Anaemia of chronic inflammatory disease

When the bone marrow cells that develop into mature blood cells and platelets (stem cells) are damaged or suppressed, the bone marrow can cease to function. This is an autoimmune disorder, in which the immune system suppresses bone marrow stem cells. The bone marrow failure leads to a dramatic reduction in RBC, white blood cells, and platelets. This condition is termed *Anaemia of chronic inflammatory disease.*

The analytes reported as part of the liver function test, in particular the serum '*Total Protein*', '*Albumin*', '*Total Bilirubin*', '*ALT (Alkaline aminotransferase)*', '*Alkaline Phosphates*' and '*GGT (Gamma-Glutamyl Transpeptidase)*' are key indicators which can be used to categorise the different types of anaemia discussed above. The ranges, aims knowledge and logic relating to these particular analytes are described in clinical guidelines which are used by clinicians to determine the cause of the anaemia, or if the anaemia is normal for the specific patient.

The implementation model is illustrated graphically in Figure 6.1. The model comprises of a LIS, a MTS system for managing messages, a AMS (*white-pages*), a DF (*golden-pages)* directory of ASK-agent service descriptions, an AR agent which acts as a repository for the agent archetype queries, a manager agent, and a group of four ASK-agents representing the '*Total Protein*', '*Albumin*', '*Total Bilirubin*', '*ALT (Alkaline aminotransferase)*', '*Alkaline Phosphates*' and '*GGT (Gamma-Glutamyl Transpeptidase)*' analyte guidelines. It is important to emphasise that the purpose of the manager agent is purely to act as a gateway between the LIS and the ASK-agent architecture. It is only used to start the validation when a set of patient data is sent to it from the LIS, and to return the report at the end of the validation deliberation. Many of these manager agents can exist on the system concurrently and they do not manage or take part in the actual deliberation activity itself. The deliberation activity is the responsibility of the group of ASK-agent's.

Section 6.2 describes the activities of the implementation model from the point of receiving a validation request from the LIS through to reporting the result of the validation service. Some further medical specific details are included to elaborate on choice decisions made by the ASK-agent on behalf of the guideline it represents.

Finally section 6.3 presents various concluding remarks.

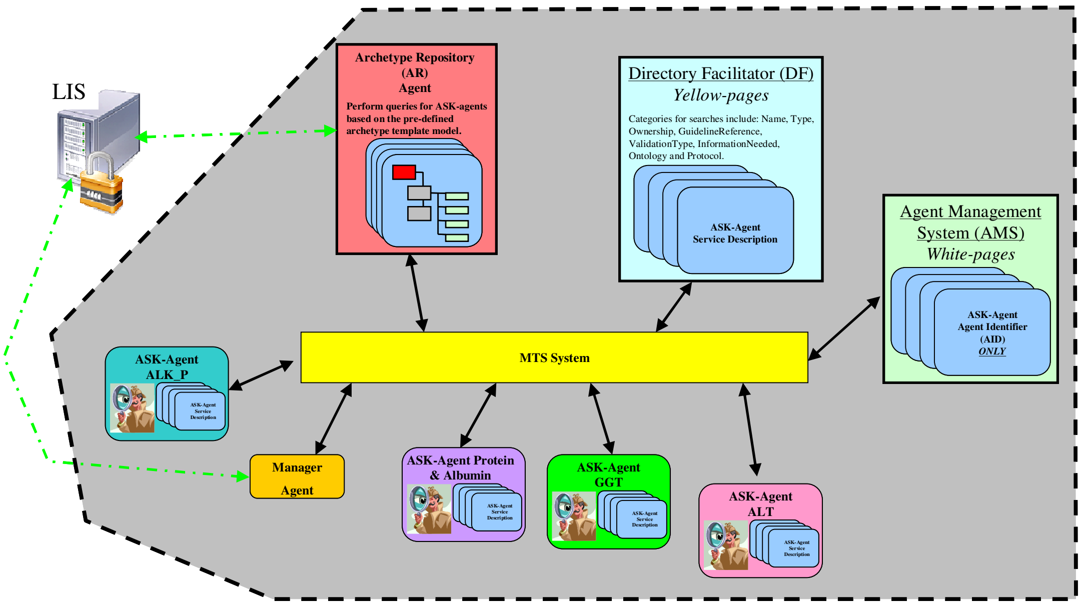

LIS
Archetype Repository (AR) Agent
Perform queries for ASK-agents based on the pre-defined archetype template model.
Directory Facilitator (DF)
Yellow-pages
Categories for searches include: Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology and Protocol.
ASK-Agent Service Description
Agent Management System (AMS)
White-pages
ASK-Agent Agent Identifier (AID) ONLY
MTS System
ASK-Agent ALK_P
Manager Agent
ASK-Agent Protein & Albumin
ASK-Agent GGT
ASK-Agent ALT

Figure 6.1 Anaemia for chronic diseases model.

6.2 Implementation model activities

The agent manager acts as a point of entry for a set of LIS laboratory results and provides a gateway for the LIS to submit validation queries into the social group of ASK-agents and coordinates the activity of starting the validation process. As many agent managers can co-exist in the system concurrently there is no bottleneck of execution. The agent manager ensures all selected ASK-agents start their execution cycle simultaneously and respond in a timely manner. The agent manager does not take part in the actual deliberation itself. All ASK-agents, whether part of the group, or not, can be contacted directly by, and have awareness of other agents through the MTS once they register with the DF service. The ASK-agent can accept incoming messages from, and send messages to, other ASK-agents according to their BDI implementation. The AR agent is encoded with one archetype query for use in this implementation, and is used to determine using the sampleID and/or patientID the gender of the patient and if the patient is female, determine if she is pregnant. The gender of the patient is used to select the appropriate reference ranges to which the results will be compared in the GGT ASK-agent, and the check for pregnancy will determine if an elevated SerumALK_P result could be due to skeletal growth related to a pregnancy. This archetype was constructed at the time of developing the ALK_P agent and reused for use with the GGT ASK-agent. There is no limit to the number of predefined archetype queries that can be placed in the AR agent.

To begin validation the LIS sends a message to the agent manager containing a set of results. To comply with standard operational procedures, a set of laboratory results are issued by the LIS for validation as a package, rather than as single result entities. This is because there are variations in processing times associated with the different laboratory tests. Validating using a group of laboratory results, such as the set of Liver Function Tests, improves the accuracy of the derived outcome, when compared to a solitary result [Fraser_b, 2000].

The structure of the LIS message payload sent to the agent manager is shown in expression (6.1) where the elements of the message payload are separated by semi-colons, but it could also be an XML, schema or object.

Validate;4121141;4;2;1.4;40;180;80 (6.1)

Where the first element is the text '*Validate*', the second element is the '*SampleID*' of the result, the third element is the '*Total Protein*' result, the fourth element is the '*Albumin*', the fifth element is the '*Total Bilirubin*', the sixth element is the '*ALT (Alkaline aminotransferase)*', the seventh element is the '*Alkaline Phosphates*' and the eight and final element is the '*GGT (Gamma-Glutamyl Transpeptidase)*', as in expression (6.2).

Validate;SampleID;T_Pro;Alb;T_Bil;ALT;Alk_P;GGT (6.2)

Therefore:	SampleID	=	4121141,
	SerumT_Pro	=	4,
	SerumAlb	=	2,
	SerumT_Bil	=	1.4,
	SerumALT	=	40,
	SerumAlk_P	=	180,
	SerumGGT	=	80.

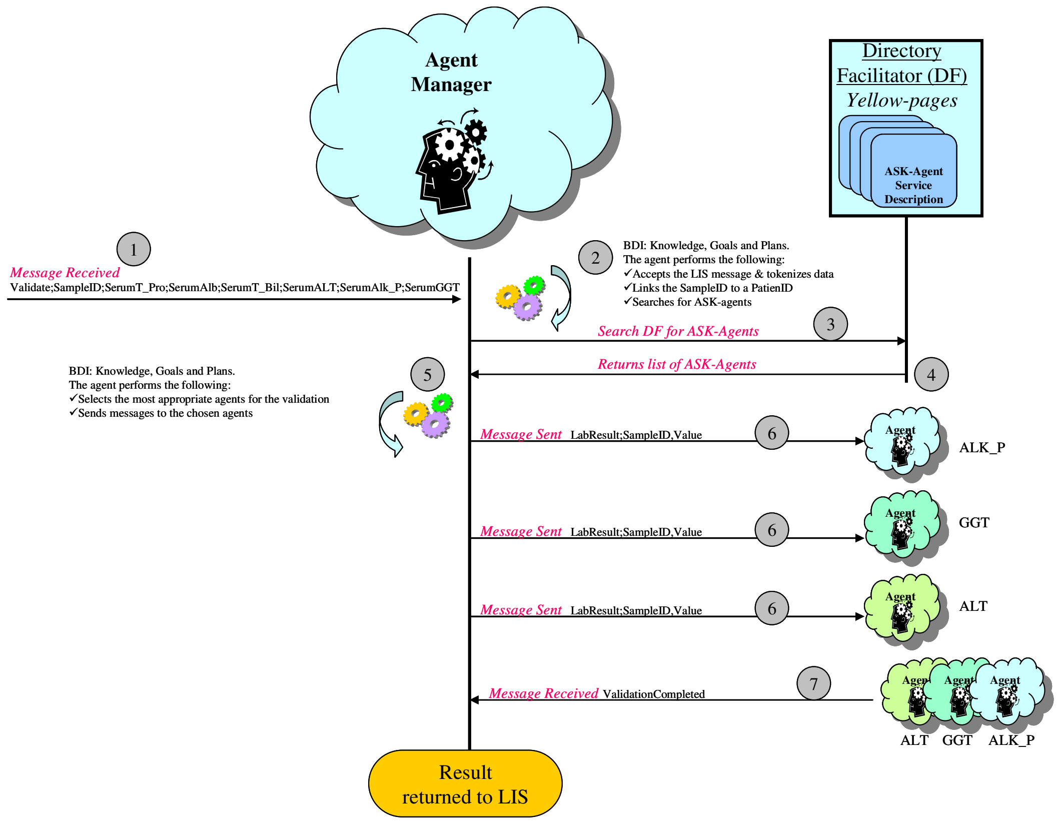
On receipt of the message the agent manager searches using the DF for suitable expert ASK-agents who are available and capable of validating the set of results or part thereof.

The following subsections describe the various agents and explain their operation. Section 6.2.1 details the Agent Manager, section 6.2.2 details the ASK-agent Protein & Albumin, section 6.2.3 details the ASK-agent ALK_P, section 6.2.4 details the ASK-agent GGT and finally section 6.2.5 details the ASK-Agent ALT.

6.2.1 Agent Manager

The function of the agent manager is to coordinate the starting activity of the agent validation process, and to act as a contact point for the LIS. The agent manager contains a belief called '*currentlyValidating*' in its belief base. This Boolean flag is set to *true* when the agent manager is currently processing a validation request, and *false* when it is idle. LIS's laboratory validation requests are queued by the laboratory system until the manager agent becomes free, which can be checked using the *'currentlyValidating'* Boolean flag. When the manager agent receives a new validation request from the LIS, it sets the '*currentlyValidating*' flag to *true,* and runs an initial plan called ManagerInitPlan.java. This plan operates for the full life cycle of the agent manager and is used to filter incoming messages.

Each ASK-agent was automatically registered with the AMS, and all its service descriptions were registered in the DF using the *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Protocol* attributes. The agent manager divides up the result set received from the LIS and searches for appropriate ASK-agents using the DF. At this point the manager agent is unaware of any other ASK-agents in the system, and the search is started in order to locate suitable ASK-agents which have the capability to validate all or part of the result set. The search returns the AID for each ASK-agent registered. The agent manager can expand the search to include any combination of the *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Protocol* attributes. The manager agent compiles a list of the various expert ASK-agents capable of validating parts of the sample result and a list of the required patient information the expert ASK-agents need. The agent manager then searches for the patient information and finds as much of the required information on the patient as possible. The manager agent compares the available information against the expert ASK-agent's requirement list and then chooses the most promising expert ASK-agents. The manager agent sends 'accept and process' messages containing information about the patients' laboratory result to the selected expert ASK-agents. A detailed control flow is shown in Figure 6.2.



Agent Manager
Directory Facilitator (DF)
Yellow-pages
ASK-Agent Service Description
1
Message Received
Validate;SampleID;SerumT_Pro;SerumAlb;SerumT_Bil;SerumALT;SerumAlk_P;SerumGGT
2
BDI: Knowledge, Goals and Plans.
The agent performs the following:
✓Accepts the LIS message & tokenizes data
✓Links the SampleID to a PatienID
✓Searches for ASK-agents
3
Search DF for ASK-Agents
4
Returns list of ASK-Agents
5
BDI: Knowledge, Goals and Plans.
The agent performs the following:
✓Selects the most appropriate agents for the validation
✓Sends messages to the chosen agents
6
Message Sent LabResult;SampleID,Value
Agent
ALK_P
6
Message Sent LabResult;SampleID,Value
Agent
GGT
6
Message Sent LabResult;SampleID,Value
Agent
ALT
7
Message Received ValidationCompleted
Agent
Agent
Agent
ALT GGT ALK_P
Result returned to LIS

Figure 6.2 Manager Agent.

Figure 6.2 Manager Agent explanation:

1) Message Received: The start of the string message begins with "Validate", which is filtered by the agent manager's initial plan, and the remaining elements are tokenized and stored as beliefs in the manager's belief base. "SampleID" is the sample's test unique ID, "SerumT_Pro" is the result total Protein, "SerumAlb" is the Albumin result which is a type of Protein, "SerumT_Bil" relates to the result Total Bilirubin, "SerumALT" is the Alanine aminotransferase, "SerumAlk_P" relates to the Alkaline Phosphates and the "SerumGGT" relates to the (Gamma-Glutamyl Transpeptidase). The units related to each attribute are standard units accepted by the medical institution, but are omitted from the proof of concept as it

distracts from the main issues. When a “Validate” order has been accepted by the manager agent it sets its *currentlyValidating* Boolean flag to true to declare it is occupied.

2) The manager accepts the message, tokenises the data and populates its belief base. Using goals and plans the manager agent links the SampleID to the PatientID. This is completed to allow the manager agent to find out more information on the patient rather than just information on the sample itself.

3) The manager agent begins to search the DF for suitable ASK-agents capable of validating this type of sample type by searching through the DF directory. Each expert ASK-agent has their services listed in the DF directory and are always registered. The main information sought is *ValidationType* and *InformationNeeded.*

4) The DF directory search returns an object containing all the DFAgentDescription’s of the agents matching the search criterion. This interaction is a form of *pull* communications, where specific information was requested.

5) At this point it is important to highlight that the DF search could be performed more efficiently, and to this end two more advanced configurations can be implemented, but are not mandatory in this implementation. The expert ASK-agent can be queried by the manager for the following information:

   (a) CEN Archetype
   Each expert ASK-agent has a CEN Archetype query capability, which was developed during the agents’ construction. The ASK-agent can perform an intelligent LIS database search for data, and the response is a context rich information format, specifically designed for processing in the agent. If queried by the manager the expert ASK-agent can provide a Boolean response indicating that all the “*InformationNeeded*” is present and available. A manager agent would not be able to choose an expert ASK-agent if all the information was not available for processing. This is a divide and conquer

approach, where the work of knowing if the expert ASK-agent has enough information to perform its validation technique rests with the ASK-agent, and the agent manager does not need to seek information on the expert ASK-agent's behalf. After completing the prototype it was considered useful to return a list of patient specific information that it could/could not locate from the Archetype. This would allow the agent manager to choose a sub-plan, which could composite missing patient information from another ASK-agent. Results from the sub-plan could be used by the original expert ASK-agent for validation.

(b) Performance queries

The agent manager can query expert ASK-agents on its performance (i.e. No. times called, No. times used, No. of successes, No. of failures). This gives the manager agent an indication of the expert ASK-agent's success rate. A manager would normally be encoded to accept a more successful ASK-agent over a less successful one.

When the manager has decided on a course of action using a plan and a list of suitable ASK-agents it then proceeds to action the plan.

6) The manager agent sends a message payload asking an expert ASK-agent to validate a specific sample using expression (6.3).

$$\text{LabResult;SampleID,Value} \qquad (6.3)$$

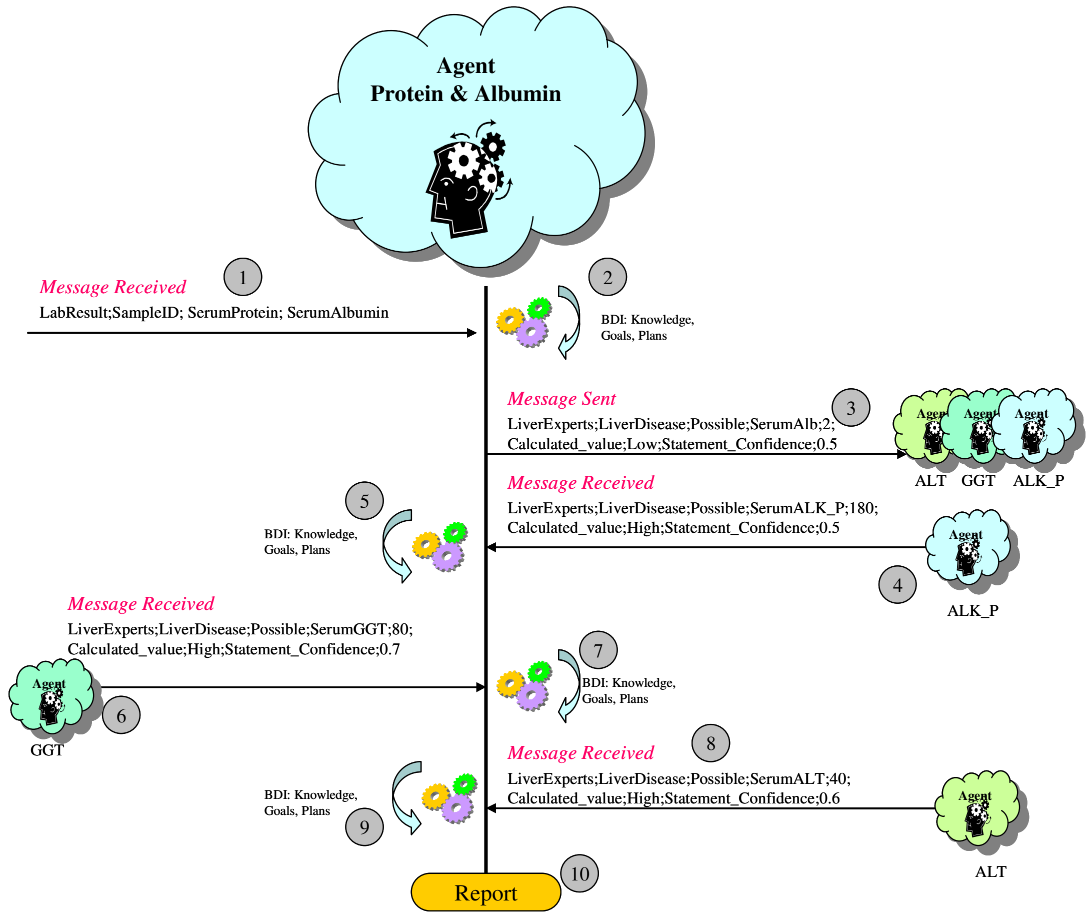
Expert ASK-agents accept the LabResult, tokenise the SampleID, the value, and populate them into their belief base and begin to validate the result.

7) When the group of expert ASK-agents have finished validating the sample result they could send a message to the manager agent declaring 'ValidationCompleted'. In the case of the proof of concept the 'ValidationCompleted' response is issued 10 seconds after the last inter-agent messages have been delivered, and the expert ASK-agents have deliberated on the result. The result is printed out on to the

computer console or could be sent to the LIS for permanent storage and the agent manager's "*currentlyValidating*" Boolean flag is reset.

6.2.2 ASK-agent Protein & Albumin

The Protein and Albumin ASK-agent implements the ASK-agent template, and expands it to include the BDI of the guideline it represents. The specific medical context of Protein and Albumin was discussed in section 2.7, and also in more detail in Appendix D. The ASK-agent receives the Protein and Albumin results from the agent manager and populates the information into the belief base. This triggers the execution of sub-plans that calculate the globulin and albumin:globulin ratio. This new calculated information is populated into the belief base of the Protein and Albumin expert ASK-agent. If the ASK-agent, during its validation cycle believes some information should be shared with other ASK-agents, it can identify a specific ASK-agent or group using the DF '*golden-pages*' service and send the information directly to them. The compulsory blackboard component allows the ASK-agent to receive and store information from other ASK-agents in the system. A detailed control flow is shown in Figure 6.3.



Agent
Protein & Albumin
1
Message Received
LabResult;SampleID; SerumProtein; SerumAlbumin
2
BDI: Knowledge, Goals, Plans
3
Message Sent
LiverExperts;LiverDisease;Possible;SerumAlb;2;
Calculated_value;Low;Statement_Confidence;0.5
Agent
ALT
GGT
ALK_P
Message Received
LiverExperts;LiverDisease;Possible;SerumALK_P;180;
Calculated_value;High;Statement_Confidence;0.5
4
5
BDI: Knowledge, Goals, Plans
Message Received
LiverExperts;LiverDisease;Possible;SerumGGT;80;
Calculated_value;High;Statement_Confidence;0.7
6
7
BDI: Knowledge, Goals, Plans
8
Message Received
LiverExperts;LiverDisease;Possible;SerumALT;40;
Calculated_value;High;Statement_Confidence;0.6
9
BDI: Knowledge, Goals, Plans
10
Report

Figure 6.3 Agent Protein & Albumin validation cycle.

Figure 6.3 Agent Protein & Albumin validation cycle explanation:

(1) The Protein & Albumin agent is running and waiting to be contacted. The agent manager sends a LabResult message to the Protein & Albumin ASK-agent containing the SampleID, Protein and Albumin results. The message received is shown in expression (6.4)

LabResult;4121141;4;2. (6.4)

Therefore:	SampleID	=	4121141
	SerumT_Pro	=	4
	SerumAlb	=	2

(2) The ASK-agent initialises an internal flag to indicate it is currently in the process of validating a patient sample result. It chooses goals and plans to complete the task.

(3) The expert ASK-agent decides to issue a message to all other expert ASK-agents of type “LiverExperts” stating it considers the SerumAlb result of ‘2’ is ‘Low’, indicating possible presence of “LiverDisease” with a “Statement_Confidence” factor of 0.5.

(4) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates the possible presence of “LiverDisease” with a confidence factor of 0.5, due to a calculated high SerumALK_P level of 180 being received from the ALK_P agent.

(5) The Protein & Albumin ASK-agent processes this information and could depending on its BDI change its direction of execution.

(6) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates the possible presence of “LiverDisease” with a confidence factor of 0.7, due to a calculated high SerumGGT level of 80 being received from the GGT agent.

(7) The Protein & Albumin ASK-agent processes this information and could depending on its BDI change its direction of execution.

(8) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates possible presence of “LiverDisease” with a confidence factor of 0.6, due to a calculated high SerumALT level of 40 being received from the ALT agent.

(9) The Protein & Albumin ASK-agent processes this information and could depending on its BDI change its direction of execution.

(10) Message passing between ASK-agents allows them to collaborate together with their combined knowledge of the sample. This ASK-agent can send messages it believes are of use to other agents. Other ASK-agents can send messages based on what they believe would be of use to this ASK-agent. Each ASK-agent can choose to use the received information in order to be guided through a possible validation direction based on evidence, or simply use the information to corroborate its own deliberation. This aspect of the ASK-agents operation is up to the guideline encoder to implement. Once finalised the report is generated.

6.2.3 ASK-agent ALK_P

The Alkaline Phosphates ASK-agent implements the ASK-agent template, and expands it to include the BDI of the guideline it represents. The ASK-agent receives the Alkaline Phosphates result from the agent manager and populates the information into the belief base. This triggers the execution of sub-plans that evaluate the result. If the ASK-agent, during its validation cycle believes some information should be shared to other ASK-agents, it can identify a specific ASK-agent or group of ASK-agents using the DF '*golden-pages*' service and send the information directly to them. The compulsory blackboard component allows the agent to receive and store information from other ASK-agents in the system. A detailed control flow is shown in Figure 6.4.

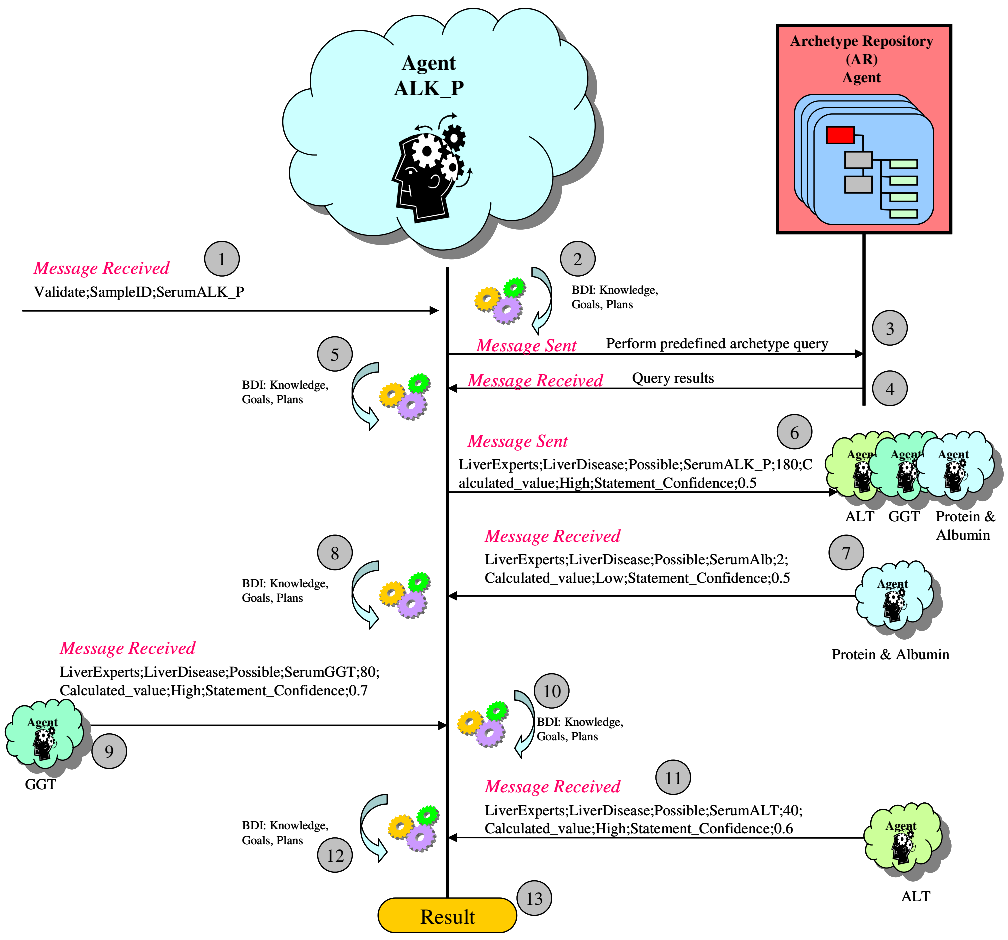

Agent ALK_P
Archetype Repository (AR) Agent
1
Message Received
Validate;SampleID;SerumALK_P
2
BDI: Knowledge, Goals, Plans
3
Message Sent
Perform predefined archetype query
4
Message Received
Query results
5
BDI: Knowledge, Goals, Plans
6
Message Sent
LiverExperts;LiverDisease;Possible;SerumALK_P;180;Calculated_value;High;Statement_Confidence;0.5
Agent
ALT
GGT
Protein & Albumin
7
Message Received
LiverExperts;LiverDisease;Possible;SerumAlb;2;Calculated_value;Low;Statement_Confidence;0.5
Protein & Albumin
8
BDI: Knowledge, Goals, Plans
Message Received
LiverExperts;LiverDisease;Possible;SerumGGT;80;Calculated_value;High;Statement_Confidence;0.7
9
GGT
10
BDI: Knowledge, Goals, Plans
11
Message Received
LiverExperts;LiverDisease;Possible;SerumALT;40;Calculated_value;High;Statement_Confidence;0.6
ALT
12
BDI: Knowledge, Goals, Plans
13
Result

Figure 6.4 Agent ALK_P validation cycle.

Figure 6.4 Agent ALK_P validation cycle explanation:

(1) The ALK_P ASK-agent is running and waiting to be contacted. The agent manager sends a LabResult message to the ALK_P ASK-agent containing the SampleID, ALK_P result as shown in expression (6.5).

LabResult;4121141;180. (6.5)

Therefore:	SampleID	=	4121141
	SerumALK_P	=	180

(2) The ASK-agent initialises an internal flag to indicate it is currently in the process of validating a patient sample result. It chooses goals and plans to complete the task. During the validation of the result the ASK-agent believes the SerumALK_P result of 180 is high.

(3) The ALK_P ASK-agent decides to request using a message that the archetype agent repository performs a predefined query to determine if the patient is pregnant. This query was configured at the time of encoding the ALK_P ASK-agent and returns a customised data set.

(4) The archetype agent repository executes the query and returns the data set back to the requesting agent using a return to sender message. The contents confirm that the patient is female and is pregnant.

(5) The ALK_P ASK-agent then alters its beliefs to reflect the returned information.

(6) The expert ASK-agent decides to issue a message to all other expert ASK-agents of type "LiverExperts" stating it considers the SerumALK_P result of 180 is high indicating the possible presence of "LiverDisease" with a "Statement_Confidence" factor of 0.5. The ALK_P ASK-agent stores that information in its local blackboard.

(7) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.5, due to a calculated high SerumAlbumin level of 2 being received from the Protein & Albumin agent. The ALK_P ASK-agent stores that information in its local blackboard.

(8) The ALK_P ASK-agent processes this information and in this case does not change the direction of execution based on the information given.

(9) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.7, due to a calculated high SerumGGT level of 80 being received from the GGT agent.

(10) The ALK_P ASK-agent processes this information and knows in its BDI that an elevated GGT is of importance and changes its direction of execution based on the information given.

(11) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.6, due to a calculated high SerumALT level of 40 being received from the ALT ASK-agent.

(12) The ALK_P ASK-agent processes this information and decides not to change its direction of execution based on the information given.

(13) The result generated at the end of the ASK-agent's execution, based on a time limit in this implementation allows for the confirmation of the diagnosis or allows all the BDI to be reprocessed.

6.2.4 ASK-agent GGT

The GGT ASK-agent implements the ASK-agent template, and expands it to include the BDI of the guideline it represents. The ASK-agent receives the SerumGGT result from the agent manager and populates the information into the belief base. This triggers the execution of sub-plans that evaluates the result. If the ASK-agent, during its validation cycle believes some information should be shared to other ASK-agents, it can identify a specific ASK-agent or group of ASK-agents using the DF '*golden-pages*' service, and send the information directly to them. The compulsory blackboard component allows the agent to receive and store information from other ASK-agents in the system. A detailed control flow is shown in Figure 6.5.

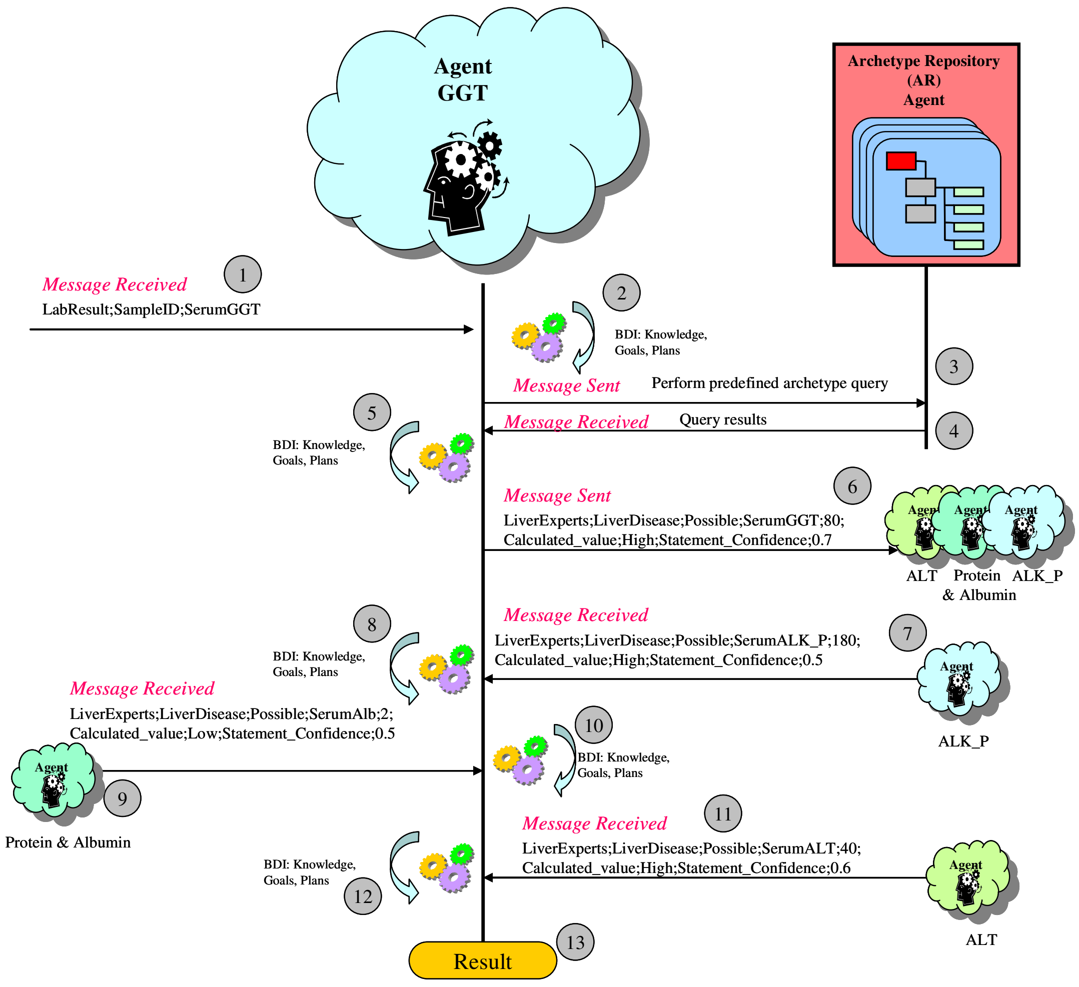

Agent GGT
Archetype Repository (AR) Agent
1
Message Received
LabResult;SampleID;SerumGGT
2
BDI: Knowledge, Goals, Plans
3
Message Sent
Perform predefined archetype query
4
Message Received
Query results
5
BDI: Knowledge, Goals, Plans
6
Message Sent
LiverExperts;LiverDisease;Possible;SerumGGT;80; Calculated_value;High;Statement_Confidence;0.7
Agent
ALT
Protein & Albumin
ALK_P
7
Message Received
LiverExperts;LiverDisease;Possible;SerumALK_P;180; Calculated_value;High;Statement_Confidence;0.5
ALK_P
8
BDI: Knowledge, Goals, Plans
Message Received
LiverExperts;LiverDisease;Possible;SerumAlb;2; Calculated_value;Low;Statement_Confidence;0.5
9
Protein & Albumin
10
BDI: Knowledge, Goals, Plans
11
Message Received
LiverExperts;LiverDisease;Possible;SerumALT;40; Calculated_value;High;Statement_Confidence;0.6
ALT
12
BDI: Knowledge, Goals, Plans
13
Result

Figure 6.5 Agent GGT validation cycle.

Figure 6.5 Agent GGT validation cycle explanation:

(1) The GGT ASK-agent is running and waiting to be contacted. The agent manager sends a LabResult message to the GGT Agent containing the SampleID and SerumGGT results as shown in expression (6.6).

$$\text{LabResult;4121141;80.} \qquad (6.6)$$

Therefore:	SampleID	=	4121141
	SerumGGT	=	80

(2) The ASK-agent initialises an internal flag to indicate to any other ASK-agent it is currently in the process of validating a patient sample result. It chooses goals and plans to complete the task.

(3) The GGT ASK-agent decides to request using a message that the archetype agent repository performs a predefined query to determine if the patient is female, as there is a considerable difference in gender based reference ranges. This query was configured at the time of encoding the GGT ASK-agent and returns a customised data set.

(4) The archetype agent repository executes the query and returns the data set back to the requesting agent using a return to sender message. The contents confirm that the patient is female.

(5) The GGT ASK-agent then alters its beliefs to reflect the returned information.

(6) The expert ASK-agent decides to issue a message to all other expert ASK-agents of type "LiverExperts" stating it considers the SerumGGT result of 80 is high indicating the possible presence of "LiverDisease" with a "Statement_Confidence" factor of 0.7.

(7) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.5, due to a calculated high SerumALK_P level of 180 being received from the ALK_P agent. The GGT ASK-agent stores that information in its local blackboard.

(8) The GGT ASK-agent processes this information and changes its direction of execution.

(9) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.5, due to a calculated low SerumAlb level of 2 being received from the Protein & Albumin agent. The GGT ASK-agent stores that information in its local blackboard.

(10) The GGT ASK-agent processes this information and changes its direction of execution.

(11) During the validation of the sample result a message is received from a "Liver Experts" group member stating this sample indicates possible presence of "LiverDisease" with a confidence factor of 0.6, due to a calculated high

SerumALT level of 40 being received from the ALT agent. The GGT ASK-agent stores that information in its local blackboard.

(12) The GGT ASK-agent processes this information and changes its direction of execution.

(13) The result generated at the end of the ASK-agent's execution, based on a time limit in this implementation allows for the confirmation of the diagnosis or allows all the BDI to be re-evaluated.

6.2.5 ASK-agent ALT

The ALT - (alanine aminotransferase) ASK-agent implements the ASK-agent template, and expands it to include the BDI of the guideline it represents. The ASK-agent receives the SerumALT result from the agent manager and populates the information into the beliefbase. This triggers the execution of sub-plans that evaluate the result. If the ASK-agent, during its validation cycle believes some information should be shared to other ASK-agents, it can identify a specific ASK-agent or group using the DF '*golden-pages*' service and send the information directly to them. The compulsory blackboard component allows the ASK-agent to receive and store information from other ASK-agents in the system. A detailed control flow is shown in Figure 6.6.

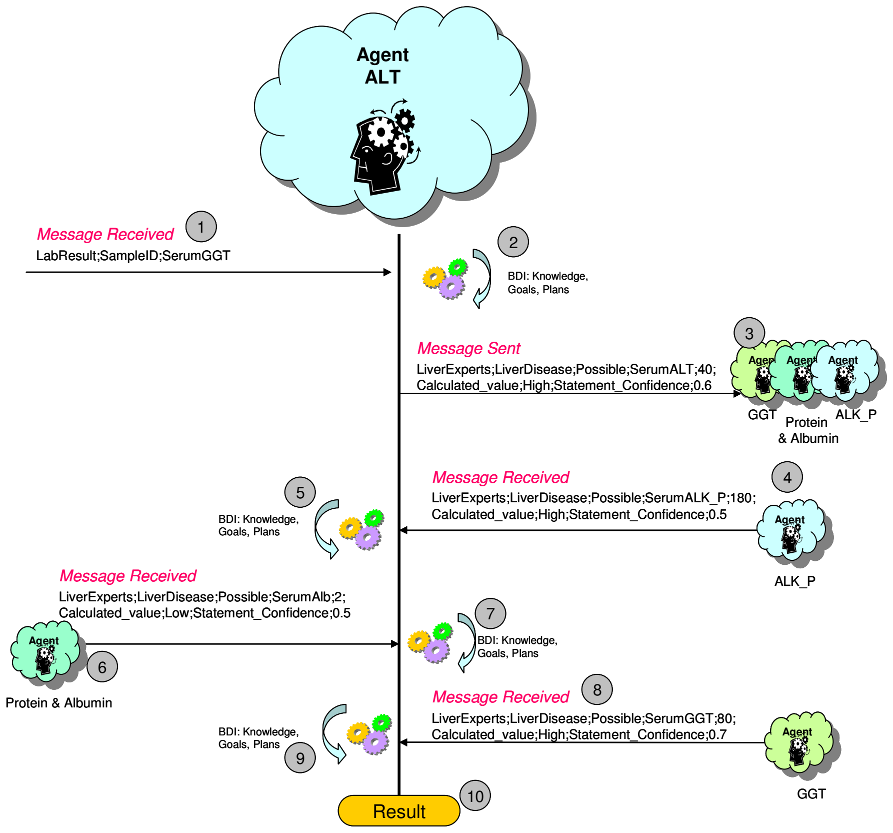

Agent
ALT
Message Received
1
LabResult;SampleID;SerumGGT
2
BDI: Knowledge, Goals, Plans
Message Sent
3
LiverExperts;LiverDisease;Possible;SerumALT;40; Calculated_value;High;Statement_Confidence;0.6
Agent
GGT
Protein & Albumin
ALK_P
Message Received
4
LiverExperts;LiverDisease;Possible;SerumALK_P;180; Calculated_value;High;Statement_Confidence;0.5
5
BDI: Knowledge, Goals, Plans
Message Received
LiverExperts;LiverDisease;Possible;SerumAlb;2; Calculated_value;Low;Statement_Confidence;0.5
6
Protein & Albumin
7
BDI: Knowledge, Goals, Plans
Message Received
8
LiverExperts;LiverDisease;Possible;SerumGGT;80; Calculated_value;High;Statement_Confidence;0.7
GGT
9
BDI: Knowledge, Goals, Plans
10
Result

Figure 6.6 Agent ALT validation cycle.

Figure 6.6 Agent ALT validation cycle explanation:

(1) The ALT ASK-agent is running and waiting to be contacted. The agent manager sends a LabResult message to the ALT Agent containing the SampleID, SerumALT results as shown in expression (6.7).

LabResult;4121141;80. (6.7)

Therefore: SampleID = 4121141,

SerumALT = 40,

(2) The ASK-agent initialises an internal flag to indicate to any other ASK-agent it is currently in the process of validating a patient sample result. It chooses goals and plans to complete the task.

(3) The expert ASK-agent decides to issue a message to all other expert ASK-agents of type "LiverExperts" stating it considers the ALT result of 40 is High indicating the possible presence of "LiverDisease" with a "Statement_Confidence" factor of 0.6.

(4) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates possible presence of “LiverDisease” with a confidence factor of 0.5, due to a calculated high SerumALK_P level of 180 being received from the ALK_P ASK-agent. The ALT ASK-agent stores that information in its local blackboard.

(5) The ALT ASK-agent processes this information and can change its direction of execution.

(6) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates possible presence of “LiverDisease” with a confidence factor of 0.7, due to a calculated high SerumAlb level of 180 being received from the Protein & Albumin ASK-agent. The ALT ASK-agent stores that information in its local blackboard.

(7) The ALT ASK-agent processes this information and can change its direction of execution.

(8) During the validation of the sample result a message is received from a “Liver Experts” group member stating this sample indicates possible presence of “LiverDisease” with a confidence factor of 0.6, due to a calculated high SerumALK_P level of 180 being received from the ALK_P ASK-agent. The ALT ASK-agent stores that information in its local blackboard.

(9) The ALT ASK-agent processes this information and can change its direction of execution.

(10) The result generated at the end of the ASK-agent’s execution allows for the confirmation of the diagnosis or allows all the BDI to be reprocessed.

6.3 Conclusion

This chapter demonstrated the novel procedure developed through this research to consistently and efficiently convert clinical and laboratory guidelines into ASK-agent modules. By combining both the ability to reproduce guideline content within a single ASK-agent, and allowing those autonomous entities to socialise and collaborate, a method of validation emerges that is based on communicating supportive information. It demonstrated that the ASK-agents at the beginning of the validation are unaware of each other, but by acting through the developed framework the expert ASK-agents were able to identify and locate other ASK-agents via the DF using *ownership* and *type* attributes, and formed a social group. Using the ASK-agent architecture, each expert ASK-agent has a representation, awareness, investigation, interaction, integration, and coordination capability with other ASK-agents.

The role of the agent manager is to coordinate activities at the beginning of the validation cycle. Once the validation cycle is initiated, the agent manager does not get involved until the ASK-agents have deliberated and returned a report. At any point in the ASK-agents validation cycle an agent can request the services of another ASK-agent. In most cases these requests are for supportive information exchanges with other ASK-agents who may not be directly involved in the validation cycle, but who can support the validation of the results by others. To perform this action the ASK-agent seeking the information can perform a DF search for the required service and select an appropriate ASK-agent. All communications are via message passing directly with the agent providing the service. The agent manager does not need to be alerted or involved in the selection of these ASK-agent requests.

Therefore, the fundamental principle of operation demonstrated here is: standalone ASK-agent entities, which were not aware of any other ASK-agents at the beginning of the validation, were able to identify, interact, and collaborate with others in order to validate a patient sample only using this novel framework. There were no inter-guideline management rules with the exception of the expert ASK-agent components, which also demonstrated ASK-agents can operate in a truly distributed fashion. This permits ASK-agents to be added, deleted, included or suspended without any effect on the processing of the system. The addition of the pull and push communications improves the collaboration capability of the ASK-agent.

Chapter

7

Results and discussion

7.1 Introduction

The function of this chapter is to present the data obtained from the proof of concept implementation, and discuss the operation of the patient-centred validation service provided by the ASK-agent architecture. The procedure to evaluate the reports produced by the ASK-agent used a real set of laboratory results for which the actual validation outcome is known in advance. This was determined through published medical documentation in the form of genuine patient case histories. This data was presented to both the ASK-agent implementation and a human expert. The human expert was Frank Clarke, Lecturer in Biological Sciences, Dublin Institute of Technology, Kevin Street, Dublin 8. His help in analysing the results as a blind test and openly discussing how he derived his conclusion afterwards was truly appreciated. Section 7.2 presents the results from a medical perspective, and compares the actual established patient medical condition to that processed by the ASK-agents. Although the implementation is performed using a single Intel Pentium 4 processor, CPU speed of 2.8GHz with 1024MB RAM the ASK-agent architecture was based on distributing the validation activity. This would allow validation to be performed on a number of separate machines distributed throughout the clinical laboratory, connected via a computer network. However, when the ASK-agent is deployed over a number of machines this could impact on network traffic. Section 7.3 presents the results in terms of predicted network traffic, and examines the scalability of the ASK-agent architecture. Section 7.4 categorises the ASK-agents contribution to the body of knowledge. Finally, section 7.5 presents various concluding remarks.

7.2 Medical perspective of ASK-agent outcome

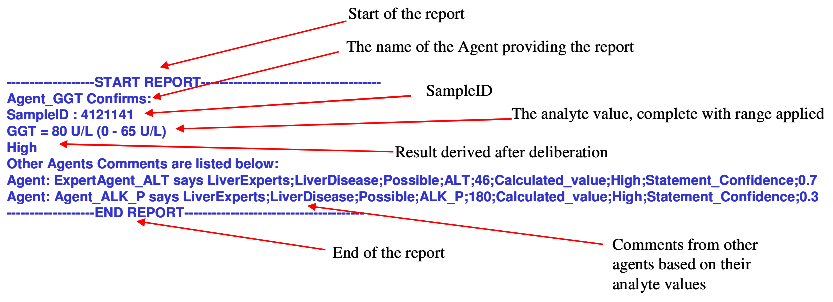
The development of the model implementation was completed in accordance with the procedures described in chapter 5. After constructing the agent plans and ADF files each ASK-agent was tested individually using test data to ensure it operated correctly and represented the guidelines content. This testing was used to verify the intra-ASK-agent operation when compared to the respective guideline. The four ASK-agents were then run concurrently and tested using additional data sets. This testing was used to verify the inter-ASK-agent operation, and ensure that the ASK-agents interpreted incoming messages correctly, and issued supportive messages to other members in the group. It was only at this stage the operation of the four ASK-agents was evaluated using genuine patient data, taken directly from documented medical case profiles with known outcomes. Each of the 10 sets of analyte results was extracted from published patient case profiles in which actual patient data, information obtained during the course of the investigation, and the interpretation of the data by the clinician was factually documented. The case histories included many more test results, and patient specific characteristics, all of which were used to establish conclusively the patient's medical state. These case profiles are key instruments used to educate and train various types of medical professionals on how to prioritise patient data, and how to interpret that data in order to make a diagnosis or manage a patient's healthcare [Marshall, 2000; Kumar & Clarke, 2002]. For the purposes of evaluating the proof of concept implementation not all the information available from the case profiles was used, and focused on the total protein, albumin, ALT, GGT and alkaline phosphates results from male patients. To ensure an unbiased interpretation of results, the human expert was not involved in the selection of the genuine patient data and was totally unaided by any additional knowledge about the patient or patient history. It is also important to emphasise that laboratory results are not normally used to diagnose a patient's condition. However, for investigative purposes in the implementation the ASK-agents disclose what conditions they would suspect the patient has if the given results are to remain plausible. In a real laboratory version of the ASK-agent system best practice would be to check the patient's healthcare records to investigate if the patient had the types of diagnosis then calculate the plausibility score, or make a statement such as "these results would be plausible for a patient with these types of diagnosis".

The selected patient results are presented in Table 7.1. This particular data emphasises standard decision-making criteria detailed in guidelines so particular paths of activity could be chosen. Some of the results are for normal patients, others can be validated using a standalone guideline, and the remaining results would require a collaboration of two or more guidelines in order to validate the results.

Patient	T Bil µmol/L	T Pro g/l	Alb g/l	ALT U/L	GGT U/L	Alk P U/L
1	22	87	46	109	221	102
2	11	85	49	36	22	51
3	12	67	37	287	179	207
4	4	66	39	21	11	73
5	14	66	30	15	23	156
6	3	37	11	27	7	29
7	11	80	48	23	14	55
8	5	80	40	40	126	140
9	17	99	54	48	199	215
10	15	65	36	84	86	57

Table 7.1 Patient result outcomes

The standard output from the ASK-agents for the purposes of the implementation is shown in Figure 7.1. The report confirms the ASK-agent, the SampleID, the analyte value complete with the applied reference range, the comment on the result after deliberation, and the comments received from other ASK-agents. It is standard practice in laboratory reporting to include the SampleID, analyte result, the applied reference range and a comment in relation to the result itself (e.g. high, low, normal etc). The addition of the other ASK-agent's comments in relation to their analyte is new and could be used by the clinician to aid in their diagnosis or selecting a number of other tests to perform on the patient. The report does not indicate how the ASK-agent handled the supportive messages from other ASK-agents, but the messages' existence in the comments section is evidence that the ASK-agent was aware of it. The handling of supportive comments from other ASK-agents is encoded into the receiving ASK-agents BDI.



```
------------------START REPORT-----------------------------------
Agent_GGT Confirms:
SampleID : 4121141
GGT = 80 U/L (0 - 65 U/L)
High
Other Agents Comments are listed below:
Agent: ExpertAgent_ALT says LiverExperts;LiverDisease;Possible;ALT;46;Calculated_value;High;Statement_Confidence;0.7
Agent: Agent_ALK_P says LiverExperts;LiverDisease;Possible;ALK_P;180;Calculated_value;High;Statement_Confidence;0.3
------------------END REPORT-----------------------------------
```

Start of the report
The name of the Agent providing the report
SampleID
The analyte value, complete with range applied
Result derived after deliberation
End of the report
Comments from other agents based on their analyte values

Figure 7.1 ASK-agent result report

The reported results from the established medical case history, the human expert and the ASK-agents are tabulated in Table 7.2. Although the comparison of the reported results between the medically established, human expert and ASK-agents interpretations are very encouraging and reasonably consistent, it is the technique by which the result is derived is the most important aspect to examine. To this end the reported results are discussed comparing the conclusion and also the technique used to derive the result. Appendix D contains a description of tests and meanings for liver function tests and can be used as additional overview references for the medical terms and measurements.

The results for patient's #4 and #7 are from healthy patients with no abnormalities in regard to the liver function tests. This was confirmed by the human expert and ASK-agent reports. Patient #2 presented a set of normal results with the exception of a mildly elevated total protein. The mild increase in total protein on its own is not indicative of any illness and can be normal for some patients. The Protein & Albumin ASK-agent sent a message to the other ASK-agents but no other comments were made. So, the consensus confirmed by the human expert and ASK-agent reports states that the result is normal. This demonstrated that the ASK-agent implementation was capable of confirming normal test results even though one result was mildly outside the reference range.

Patient	Established	Human Expert	ASK-Agents
1	Acute liver disease most likely alcohol induced.	Liver disease, drug induced as a possible cause (alcohol can be one of the drugs). No time dimension chronic or acute reported.	Liver disease confirmed by the ALT ASK-agents confidence 0.8. Alcoholic disease also confirmed by the GGT ASK-agents confidence 0.7. No time dimension chronic or acute discussed.
2	Normal	Normal, but the mildly elevated total protein was noted but eliminated as it was not excessive.	Normal, the Protein & Albumin ASK-agent commented on the elevated protein, but the consensus amongst ASK-agents was a normal result.
3	Chronic liver disease most likely alcohol induced	Liver disease, specifically alcohol induced as a possible cause. No time dimension chronic or acute reported.	Liver disease confirmed by the ALT, GGT and ALK_P ASK-agents confidence 0.9. Alcoholic disease also confirmed by the GGT ASK-agent confidence 0.8. No time dimension chronic or acute reported.
4	Normal	Normal.	Normal, No additional comments.
5	Patient with a chronic skeletal disease	Not liver based disease. Suggest bone or gastrointestinal tract (GIT) based disease. But could also be improper handling of specimen or delay.	Elevated ALK_P reported by the ALK_P ASK-agent but no elevated GGT reported. Bone or pregnancy related as possible cause reported by the ALK_P ASK-agent. Gender check returned Male so pregnancy eliminated. Possible Bone disease reported confidence 0.6.
6	Malnutrition	Most likely chronic malnutrition, perhaps anorexic, or a stage of chronic liver disease.	Protein & Albumin ASK-agent reported malabsorption with confidence 0.6 and dehydration with confidence 0.3.
7	Normal	Normal.	Normal, No additional comments.
8	Medication induced liver disease, side effect of cancer treatment/therapy drugs.	Liver disease, drug induced as a possible cause.	Liver disease confirmed by the GGT and ALK_P ASK-agents confidence 0.8. Alcoholic disease also confirmed by the GGT ASK-agent confidence 0.8.
9	Bone marrow disease (This inhibits the reprocessing of aged blood products, red blood cells, globulin, etc).	Very difficult results to interpret but would suggest *Hypergammaglobinaemia*[13], some type of bone related disease reducing the reprocessing of aged blood.	Protein & Albumin ASK-agent reported possible myeloma with confidence 0.5, and also reported possible blood production disease with confidence 0.7. Liver disease suspected by the GGT and ALK_P ASK-agents confidence 0.5.
10	Liver disease, drug-induced hepatic injury caused by medication to fight Hepatitis.	Liver disease, drug induced as a possible cause.	Liver disease confirmed by the ALT ASK-agent confidence 0.9. (Lack of excessive GGT eliminates alcohol being the cause)

Table 7.2 Patient result outcomes

[13] *Hypergammaglobulinemia* – is an excess of gammaglobulin in the blood. It occurs in chronic inflammations, chronic bacterial infections, liver disease and multiple myeloma.

The results for patient #6 presented a reduced albumin and total protein with other results being normal. The Protein & Albumin ASK-agent reported malabsorption with a confidence of 0.6 and possible dehydration of 0.3, to all other agents. Although malnutrition was the established result the reported malabsorption indicates a difficulty in absorbing nutrients (which includes protein) from food, and the reported dehydration could be from a patient with diarrhoea. Both of these reported indications would support the established malnutrition result as they hamper the absorption of protein by the body, but the guideline did not contain a direct reference to malnutrition. The human expert reported similar findings which included chronic malnutrition, anorexia or a stage of liver failure. This demonstrated that the ASK-agent implementation could derive a conclusion based on a single ASK-agent as no other supportive information was received to aid in the conclusion.

The result for patient #5 presented an elevated ALK_P which was reported by the ALK_P ASK agent, but no supportive information from the GGT ASK agent was submitted to indicate liver disease. An elevated ALK_P on its own is used extensively as a marker for tumours, but can also be related to bone injury or pregnancy. The ALK_P ASK-agent queries the patient's data and identifies the patient as a male, so the pregnancy aspect was eliminated and a possible bone disease was reported with a confidence factor of 0.6. This demonstrated that the ASK-agent implementation was deriving a conclusion based on a single ASK-agent, which was able to set a goal to check the gender of the patient using an archetype query, thereby altering its course of activity and changing its reference limits if required.

The results for patient's #1 and #3 are from patients with alcohol induced liver disease with the former being an acute case and the latter a chronic case. Separately the GGT and ALK_P ASK-agents in themselves are not capable to definitively confirm the presence of liver disease. But they can issue supporting messages to each other to confirm its presence. In the case of patient #1 the ALT ASK-agent reported liver disease due to the elevated ALT, and GGT ASK-agent reported alcoholic disease due to the elevated GGT. In the case of patient #3 liver disease was reported twice (a) by both the GGT and ALK_P ASK-agents and (b) by the ALT ASK-agent. The presence of two or more factors confirming the same conclusion can be used to increase the confidence factor of the group of agents. The elevated GGT for patient #3 also confirmed alcoholic disease by the GGT agent. This test demonstrated that ALK_P and GGT ASK-agents collaborated on deriving a result, and the latter set (i.e. for patient #3) in particular was also vindicated independently by the ALT ASK-agent.

The result for patient #9 presented an elevated ALK_P and GGT which was reported by the ALK_P and GGT ASK-agents that liver disease was present confidence 0.5. The Protein & Albumin ASK-agent interpreted the elevated total protein and very mildly elevated total biliary as possible myeloma with confidence 0.5, and possible blood production disease with confidence 0.7. This finding is not uncommon because a chronic inability of the body to process aged blood components can cause liver disease. So the lack of correct blood production would be considered the main cause with liver damage being a side-effect. This demonstrated that the ASK-agent implementation was capable of accepting one or more possible outcomes simultaneously. In this case the confidence factor could be used to prioritise the derived conclusion with the blood production disease with confidence 0.7 being the larger of the reported conditions, but this is not always the case.

The result for patient #10 presented an elevated ALT indicating the presence of liver disease but the lack of increased GGT eliminates alcohol as being the cause. Although both the human expert and ASK-agent determined the presence of liver disease neither established Hepatitis as being the cause. Determining the presence of Hepatitis is completed by checking for the presence of antibodies using biology/immunology testing. The results presented did not contain any reference to antibodies being present so no comment either by the human expert or ASK-agents in relation to Hepatitis could be made. This demonstrates the ASK-agents implementation's ability to only use the information known about the patient in order to derive a result. It was not designed to definitively conclude the cause of the problem based on the patient's results, only share supportive information in which a clearer decision can be made based on the available evidence.

The result for patient #8 presented an elevated GGT and ALK_P. Both the GGT and ALK_P agents communicated with each other the elevated results and jointly confirmed the result was indicative of liver disease. But the elevated GGT level also triggered the GGT ASK-agent to report presence of alcoholic disease which was not factually correct. On consultation with the human expert it was confirmed that an elevated GGT can be a side-effect of some cancer therapy drugs and was the rational behind the human expert using the term “drug induced”. As the ASK-agent implementation was not encoded with this information it concluded that the elevated GGT was simply a result of excessive alcohol consumption. This set of results illustrated that the strength of the ASK-agents reported conclusion directly relates to the accuracy of the encoders interpretation of the guideline.

Taking into consideration that the ASK-agents were encoded with reference only to the guideline and nothing else, and that the presented results were only an extract of the full patient’s diagnosis, the reported results compared favourably with the human expert and established medical history. In addition to the reported medical results themselves, the operation of the ASK-agent architecture demonstrated that an ASK agent can operate in a standalone capacity, collaborate with other ASK-agents and conclude a number of alternative verdicts which can be observed in the data simultaneously.

7.3 ASK-agent network transmission overhead

The ASK-agent architecture is based on establishing a number of autonomous self-contained knowledge sources which communicate with each other to validate patient results. For this activity to be realised it is necessary to quantify the impact such an architecture would have on a computer system. With more of the original guideline being encoded using the ASK-agent approach it is expected that the size of the application will also increase. As the ASK-agent architecture is based on the distributed processing model, the processor loading is of minor concern, because separate processors can be utilised in order to give the desired performance. However, as the ASK-agents communicate with each other via messages, it is important to quantify the impact the proposed ASK-architecture will have on the network supporting the computer system. The act of communication between ASK-agents is supported via three primary components: the communication facility, the message envelope and message payload, as illustrated in Figure 7.2. Each of these components has an effect on the message length and/or speed at which the information can be transmitted from one location to another.

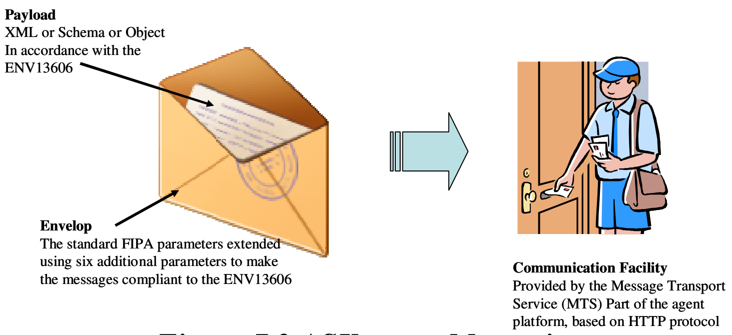

Payload
XML or Schema or Object
In accordance with the
ENV13606
Envelop
The standard FIPA parameters extended
using six additional parameters to make
the messages compliant to the ENV13606
Communication Facility
Provided by the Message Transport
Service (MTS) Part of the agent
platform, based on HTTP protocol

Figure 7.2 ASK-agent Messaging.

Communication Facility

The communication facility is provided by the base-agent message transport service (MTS) which utilises the network system [Helin & Laukkanen, 2002]. The network system is the physical transmission of data from one location to another. Although a number of high-speed LAN technologies are available, the leading market choice is the Fast Ethernet or 100BASE-T network which has been standardised as *IEEE 802.3*[14] [GEA 1998]. Although the network transmission rate is benchmarked at 100Mbps it is not possible for the network to guarantee that the communication of customer data will be 100 per cent efficient [Koes et al., 2004; GEA 1998]. Inefficiencies on the network are mainly due to packet queuing delays at routers, switches, and packet crashes or corruption which require messages to be resent. Taking a pragmatic approach to network traffic it has been estimated that on average 10bits of network transmission are required to transmit 1 byte of information. This data is based on the intra-network traffic (not internet traffic) within the Dublin Institute of Technology-Kevin Street campus, which could be considered similar from a computer network perspective (similar number of PC's and users) to a medical hospital. The MTS resides on top of the network and manages the communications between ASK-agents.

[14] *IEEE 802.3* is a collection of IEEE standards defining the various layers such as physical, media access control (MAC), data link layer of the internet [http://grouper.ieee.org/groups/802/3/].

Message envelope

The length of the FIPA message envelope is determined by the presence of the number of mandatory and optional parameters transmitted as part of the message. If a consistent set of parameters is established for all messages then the envelope length will be fixed [FIPA, 2002]. In section 4.4 it was shown that in order to make the FIPA message envelope compliant to the CEN ENV 13606-4:1999 medical communication standard it would require an additional six user defined parameters to be added to the existing FIPA envelope. The addition of these components to the mandatory FIPA parameters increases the fixed length of the message envelope to 65,536bytes.

Message payload

The length of the payload will not be fixed and depends on the quantity of information being transmitted, such as an XML document, schema, or an object. In the case of the implementation only short message payloads are being used to communicate between ASK-agents so the output can be viewed at a console level by the operator. These payload lengths are acceptable for the transmission of supportive information between ASK-agents, but would not provide a representative sample of files containing Electronic Healthcare Record (EHR) extracts or archetype queries. These extracts are documents which capture a particular view of the record and store the information in order to preserve the context so it can be communicated to another entity. On reviewing 18 randomly selected XML schema examples it was observed that the files ranged in size from 1,176bytes to 354,806bytes [available at http://www.openehr.org/; www.centc251.org]. Where the smaller size relates to a single value and the larger size relates to multiple nested groups of values. The large size of XML documents is also due to the verbosity of the label names capturing the structure.

To quantify the impact the ASK-agent architecture would have on the network, the number of messages between the agents was counted, and the duration for the validation to take place was determined using the internal PC clock and shown in Figure 7.3. This data is based on the 10 patient results processed by the ASK-agent as shown in Table 7.1. It confirms that the average number of messages being sent for one validation process to take place among the 5 agents is ≈17 messages, and the average duration for each validation cycle is 1.44 seconds. This equates to ≈12 full messages being sent per second.

Patient	ALT	Protein & Albumin	ALK_P	GGT	Manager	Total	Time Sec
	# Message	No. Message	No. Message	No. Message	No. Message		
1	4	1	1	4	5	15	1.2
2	1	4	1	1	6	13	1.4
3	7	1	7	8	5	28	1.4
4	1	1	1	1	5	9	1.5
5	1	1	8	2	6	18	1.7
6	1	8	1	1	5	16	1.5
7	1	1	1	1	5	9	1.4
8	1	1	4	8	6	20	1.3
9	1	8	4	5	5	23	1.4
10	7	1	1	1	6	16	1.6
					Average process time in seconds	1.44	
					Average Number of Message	16.7	
					Standard Deviation	5.926	

Table 7.3 ASK-agent message schedule

If it was assumed the ASK-agents were located on individual machines it is possible to predict the network transmission demand for the ASK-agent architecture for varying sizes of message payload. Table 7.4 shows the expected network traffic which would be encountered if 12 messages are being transmitted per second, with a standard envelope length of 65,536bytes, and the varying sizes of message payload ranging from 2,000bytes to 360,000bytes. This table shows the effect an increasing size in message payload would have in regard to the existing validation procedure being performed using a standard network (with an efficiency of 10 bits per byte/seconds) in a comparable amount of time to the standalone implementation.

Variable payload size (bytes)	2000	5000	8000	15000	50000	100000	150000	250000	360000
Fixed envelope size (bytes)	65536	65536	65536	65536	65536	65536	65536	65536	65536
Total message size (bytes)	6.75E+04	7.05E+04	7.35E+04	8.05E+04	1.16E+05	1.66E+05	2.16E+05	3.16E+05	4.26E+05
No of Messages	12	12	12	12	12	12	12	12	12
Size of data bytes transmitted/second	8.10E+05	8.46E+05	8.82E+05	9.66E+05	1.39E+06	1.99E+06	2.59E+06	3.79E+06	5.11E+06
Network inefficiency 10bits per byte/seconds	8.10E+06	8.46E+06	8.82E+06	9.66E+06	1.39E+07	1.99E+07	2.59E+07	3.79E+07	5.11E+07

Table 7.4 Payload size to network transmission rate

These network transmission rates are illustrated graphically in Figure 7.3 where the lower and upper dashed lines indicate the 10Mbps and 100Mbps network benchmarks respectively. Figure 7.3 illustrates that if the existing setup of ASK-agents was distributed on a network which interconnected the five machines, and the message payload length was 2,000 bytes, it would require approximately 8.1Mbps of network bandwidth to transmit the required messages to the five agents. But if the message payloads were considerably larger, such as 360,000bytes the required network bandwidth would increase to 51Mbps to maintain the existing performance.

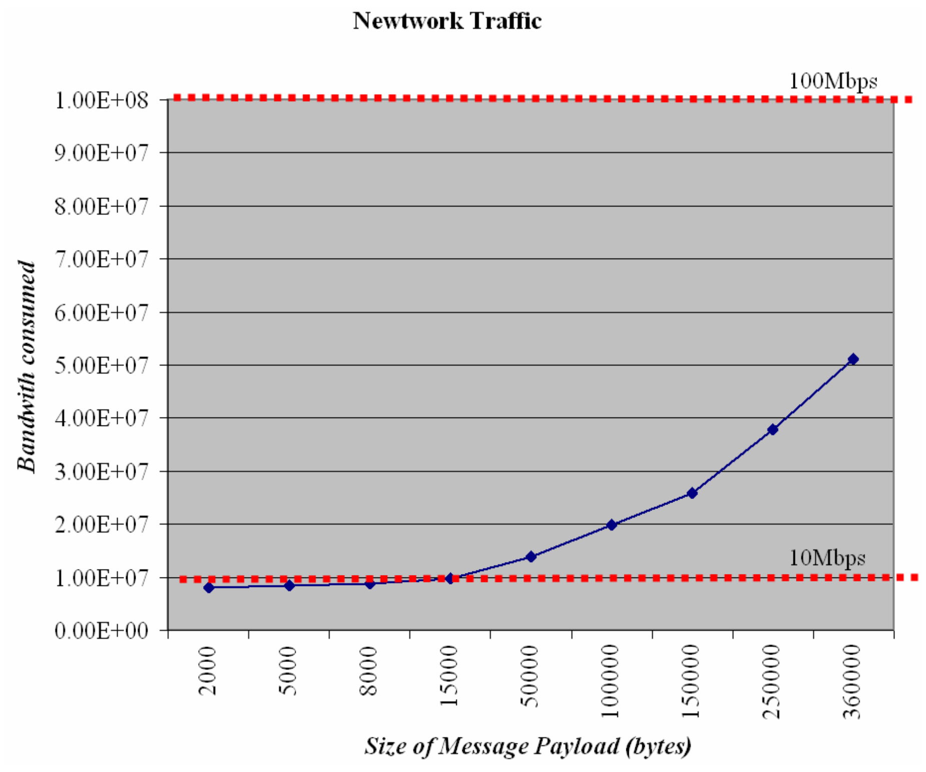


Figure 7.3 Bandwidth consumed to Payload size

This example illustrates that the sharing of supportive information between agents depends on the message size, time available and method of communication (network or PC based messages). There are various techniques which can be used to reduce the network bandwidth consumption, which are:

(1) The FIPA message protocol supports the use of a bit-efficient representation, which offers a 4:1 ratio saving in message size [FIPA, 2002; Helin & Laukkanen, 2002].

(2) The positioning of ASK-agents with a high degree of coupling, from a message perspective, with other ASK-agents could be located on the same machine in order to minimise the requirement to transmit messages over the network.
(3) Rationalising the amount of information that needs to be transmitted between ASK-agents [Helin & Laukkanen, 2002].
(4) Permitting the validation to be performed over a longer period.

Therefore, if the ASK-agents are implemented using separate machines then the network transmission overhead would be a limitation, but if the agents were selectively grouped on to a small number of machines the overhead will be minimal.

7.4 Delivery of knowledge contribution

Although the main objective of both the CIG and ASK-agent approaches are to validate patient results the modus operandi by which they provide the service are dramatically different. In section 1.3 a list of seven key objectives were presented that act as a benchmark for the comparison of the ASK-agent and the existing CIG approaches. To demonstrate the true advancement the ASK-agent offers it is important to compare each objective in turn against existing approaches.

7.4.1 Preserving the original motivation

This novel architecture provides a means by which motivational components of a clinical or laboratory guideline can be captured and encoded into a standalone ASK-agent module as illustrated in Figure 7.4. The ASK-agent architecture permits all ASK-agents to inspect the beliefs, desire and intention of other ASK-agents which are accessible through the DF registered services. The development of a tangible motivational/desire component provides representative, investigative and awareness aspects to the ASK-agent, which supports the activities of interaction, integration and coordination between the separate ASK-agent modules. This is not the case when using traditional CIG methods for encoding the guideline motivation, knowledge and logic, where the motivation/desire component is absorbed into the central management engine of the CIG. When a new guideline is decomposed into computer based symbols, elements such as desire are linked and nested through the control logic of the CIG, thereby creating an ever increasing list of centralised control logic rules.

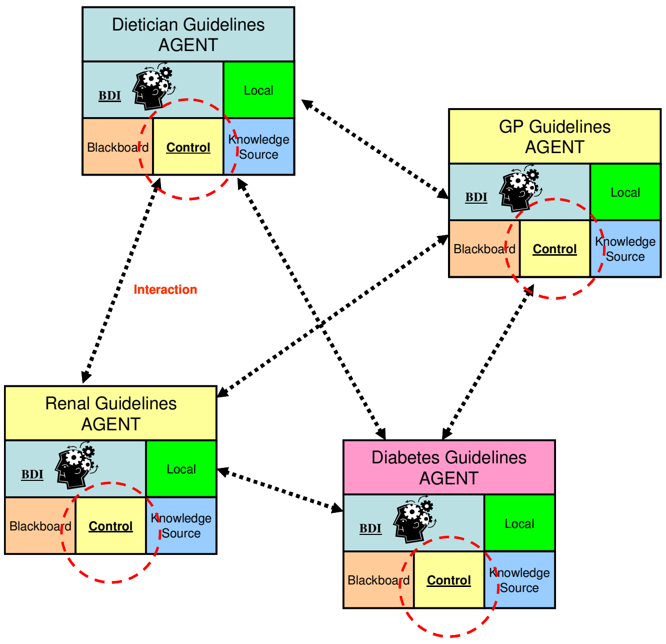


Figure 7.4 Agent architecture individual motivation/desire

7.4.2 Collaboration of knowledge sources

The existing CIG architecture solution for providing collaboration between knowledge sources (KS) operates through the use of a blackboard, where the blackboard system provides a centralised repository for showing information and provides control switching to suitable KS's. The use of a centralised blackboard and control logic allows for collaboration between the KS's to take place. This collaboration approach has a defined outcome for specific choices made during its execution. "Collaboration" software is different from "collaborative" software with the latter being executed *on the fly* (specific choices made only as they occur) and the end result cannot be truly predicted. In chapter 3 six key collaboration challenges developed by Corkill were introduced, namely; representation, awareness, investigation, interaction, integration and coordination. Table 7.5 compares each of Corkill's six key challenges whilst being applied using the blackboard and the ASK-agent approach. The outcome from the comparison provided in Table 7.5 is that the ASK-agent introduced in this thesis has addressed more of Corkill's challenges than the existing blackboard approaches when providing a collaboration framework.

But as discussed in section 3.7 Corkill's six key challenges were aspirations of collaborative software. Collaborative activity from a social viewpoint showed there was a classic separation of action from activity. This concept would permit the ASK-agent to pursue activities that may not lead to the goal of result validation, but through its action support the validation of the result by other ASK-agents, thus forming a collaboration. For example, an action by ASK-agent requiring blood glucose to validate a laboratory sample could use a recent and acceptable result of urine analysis, to give a related value allowing the validation procedure to continue. This activity would provide a method where the patient information customises the validation procedure making the system patient-centred.

Requirements	Existing CIG System/Blackboard		ASK-agent	
Representation	• Inter-KS communication not permitted. • A KS cannot store information about other KS's.	×	• DF stores attributes: *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Protocol.* • ASK-agents can identify other agents and query them directly in order to create a representation of one another	√
Awareness	• Provides a common area where KS's view current information. • The blackboard control engine is responsible for selecting the most appropriate KS and switch control to them.	√	• ASK-agent identifies an agent or group of agents, through DF and can communicate directly with them. • Localised blackboard in each ASK-agent, used to provide agent with awareness of other ASK-agents and their activities.	√
Investigation	• A KS cannot inspect goals or activities of other KSs. • Each KS has access to the blackboard only.	×	• Ontologies are used to allow ASK-agents accurately describe their current activity or goals to any other agent. • Localised blackboard used to store other agents communications, activities and goals.	√
Interaction	• Blackboard engine supports KS's to complete concurrent work under limited conditions. • No KS can interact with another KS outside the control of the blackboard.	×	• An ASK-agent acts autonomously through the execution of its beliefs, desires and intentions, and after being instantiated it continuously operates. ASK-agent has the ability to work on shared tasks.	√
Integration	• Results produced by a KS are returned into the common area of the blackboard. Therefore KS outcomes are available to all KS linked to the inference engine. But the switching of the control is still the responsibility of the inference engine.	√	• An ASK-agent receiving information can update its local beliefbase. If the information is of importance to the ASK-agents operation it could provide alteration to its execution. If two or more ASK-agents send messages the results could be combined within the receiving ASK-agent and again alter the execution.	√
Coordination	• The blackboard inference engine ensures KS modules focus their activities in the same general direction. The inference engine has full control of the direction of activity.	√	• ASK-agent has subjective coordination (intra-agent dependencies) and objective coordination (inter-agent dependencies) approaches. The intra-agent coordination is provided by the guideline where its expert knowledge is encoded to produce the agents' autonomous action, and the inter-agent coordination interface is provided by virtue of the overlapping domain knowledge. • Overlapping domain knowledge management is realised through DF and the compulsory set of beliefs and actions. These components specify and enact social laws to achieve global coherent behaviours.	√

Legend: × - the requirement is not available or is not adequate.

√ - the requirement is available or is adequate

Table 7.5 Collaborative action comparison of blackboard and ASK-agent

.

7.4.3 Accessibility to ASK-agent logic and knowledge

Existing CIG systems do not permit direct communications to be established between KSs and any other resource, except the inference engine. This means that the expert knowledge is only available when the inference engine believes it is necessary to use this resource. This places an access barrier between the KS and the resource trying to utilise it.

The ASK-agent on-the-other-hand is based on wrapping the knowledge source element into an autonomous and standalone module, and providing an access portal, using message passing, through which other resources can access it. There is no obstacle between the knowledge source and the resources trying to utilise it, except the ontology of the message being used. This ontology simply ensures that the understanding of the sender is the same as that provided by the listener.

7.4.4 Support the activities of clinicians and laboratory technologists

As discussed in chapter 1, the purpose of a laboratory result is to provide supportive information to the clinician in order for them to make a diagnosis with a degree of certainty. The role of the laboratory technologist in the scheme of this healthcare delivery is to examine the result before issuing it to the clinician, to ensure the result is plausible for the specific patient [Westgard_d, 2002]. This is because the reporting of incorrect or inaccurate sample results could have a detrimental affect on patient health. It was shown that most laboratory tests have an index of individuality <1.4 making the results of little utility for screening or diagnosis unless some stratification is in place (i.e. gender, age, disease). The clinician will have additional patient information and can decide themselves if the result of the test shows disease in their specific patient. However, under certain circumstances not all the necessary information is available to the laboratory staff in order to validate the result in accordance with the standard laboratory workflow as provided by the existing CIG systems. In these cases the laboratory technician needs to use alternative validation techniques in order to ensure the sample result is plausible [Westgard_d, 2002] (see section 3.8).

Access to the ASK-agent information is via message passing only, so other resources, whether it is another agent, human or otherwise can query the ASK-agents knowledge to gain the alternative complementary or assistive information in order to support a decision. Therefore the proposed framework allows the most appropriate supporting information to be made available aiding the clinician and laboratory technologist to deliver patient-centred healthcare, without having to process the results through the manager agent.

7.4.5 Add, delete or alter modules without system reconfiguration

With the large amount of clinical and laboratory guidelines in current use, the housekeeping required to manage these resources is an unwieldy task [Peleg_d et al., 2003]. Any architecture managing these guidelines must therefore provide a facility for altering, adding and deleting guidelines.

The blackboard system incorporates a centralised blackboard and control logic inference engine with a selection of external KS's. The cycle of execution begins with information being placed on the blackboard. The inference engine then chooses and switches control to the most appropriate KS [Corkill, 2003]. When a KS is added or altered the inference engine needs to be transformed to include the resource, but also changed to manage how and when the rule is used to avoid conflicts with other guideline KS. The deletion of a KS would also have a similar impact to the inference engine requiring the deletion or altering of control logic management rules in order to operate in the desired way. Therefore, the blackboard system provides the facility for altering, adding and subtracting guidelines, but its architecture makes it difficult to implement.

The ASK-agent is designed so that each ASK-agent is self-contained. The ASK-agent uses its belief, intention, and formally expressed desire to reproduce the decision-making potential provided through the guidelines. The ASK-agent also includes a local blackboard where messages from other ASK-agents are stored and used to guide and influence the agent's execution. Each ASK-agent is aware of other agents through the DF search/lookup service and can query possible collaborative partners directly. This self-contained technique makes it possible for ASK-agents to be added or deleted without any alteration being required to the system. The lack of centralised control diminished the management conflicts when the group of agents is altered. The output from the group depends on the makeup of its members. Therefore if a possible member of the group is absent (e.g. perhaps currently working for another validation service), then the outcome may be different to that if the member was included. The ASK-agent architecture permits multiple instances of the same ASK-agent to be created (see section 5.2) and multi-threading of these instances is managed by the agent platform. However, due to network and processing resource limitations it may be necessary to limit the number of ASK-agents of a particular type being created simultaneously.

7.4.6 Grouping ASK-agents to improve index of individuality

The index of individuality (II) plays an important role in quantifying how applicable standard population based reference ranges are to certain laboratory tests. It was shown in Table 2.10 that most of the laboratory tests presented have marked *individuality* (i.e. a low index of individuality), which minimises the usefulness of the references for general screening and testing unless stratification takes place. In addition to this, laboratory tests with an II <1.4 are of little use on their own, especially in identifying a significant change in a result or in a patient [Collier, 2001; Fraser_b, 2000]. However, the outcome of these tests can be improved by stratifying the population into homogeneous sub-groups, if they are available, thereby increasing the II which improves the *signal-to-noise ratio*[15] of the patients results [Petersen, 2005].

The proposed ASK-agent advances the Index of Individuality for groups of laboratory tests in two distinct ways:

(1) Stratification using searchable expert ASK-agents

The ASK-agent architecture supports the accurate searching and selecting of validation services through the DF. The services available are compared to the known patient information in order to identify the most appropriate validation course of action.

[15] The *signal-to-noise ratio* refers to the ratio of useful information to false or irrelevant data.

(2) Communication between groups of expert ASK-agents

If a single ASK-agent is not identified using the DF, the system can select a group of ASK-agents that can collaborate in order to validate the result. The agency approach permits separate autonomous agent modules to communicate supportive information between group members and collaborate on the validation of a result.

7.4.7 Evaluate the operation of ASK-agents

The proof of concept model as demonstrated in chapter 6 and the results presented in section 7.2 and 7.3 were used to evaluate the operation of the ASK-agent system. The evaluation was based on using genuine patient data which was extracted from case histories with known outcomes. This evaluation showed that although the execution of the ASK-agents is different to that used by existing CIGs, the derived results obtained were very similar to that of the known case history and that of the human expert. In addition to the medical result comparison, the evaluation also demonstrated that an ASK-agent could operate in a standalone capacity, in groups, collaborate with two or more ASK-agents and conclude a number of alternative verdicts which can be observed in the data simultaneously.

While performing this evaluation a number of technical issues where also reported. The additional encoding of the guideline document used by the ASK-agent increases the size of the application being developed. This is because the CIG approach does not implement all the guideline content, only selected parts due to processing complexities. This means that encoding a number of guidelines using the CIG system would have a smaller encoding footprint when compared to the same guidelines being encoded using the ASK-agent approach. However, a main feature of the ASK-agent architecture is it has the ability to perform distributed processing. This means the execution of ASK-agents can be distributed to economise on processor usage. A draw back of this is the ASK-agents reliance on message passing in order to communicate. It was shown that as processing becomes progressively more distributed through out a number of computers the network traffic overhead to permit communication begins to increase. But if the agents were selectively grouped on to a small number of machines the overhead will be minimal.

7.5 Conclusion

The implementation demonstrated that guideline motivation, knowledge and logic are presented in a format that can be coded within a standalone ASK-agent. When these separate ASK-agents are incorporated together using the novel architecture they can interact in order to validate patient samples. An important limitation of the ASK-agent is its reliance on the overlapping knowledge between the separate standalone agents. From a medical perspective this overlapping knowledge was available by selecting a medical domain in which a number of different guidelines are available. The results from processing the genuine test data through the implementation showed that the ASK-agent approach offers a plausible solution to validating laboratory results. Although the modus operandi by which the results were derived is different from that provided by traditional CIG systems the outcomes are very similar to that provided by the human expert and the case history. In producing these results the ASK-agent approach did not use a centralised set of management rules, but instead utilised a group of standalone modules.

The ASK-agent architecture can be implemented in a number of ways each having various effects on the computer system and network. If all instantiations of the ASK-agents were located on individual machines the increase in network traffic would be considerable, but local processing demand would be minimal. However, by selectively grouping the ASK-agents and managing their interactions this network traffic overhead could be dramatically reduced and an optimal processing demand for the distributed machines could be reached.

The ASK-agent architecture is based on providing an environment by which individual modules can be used to perform patient-centred validation without the need to provide a centralised set of management rules. However, from a practical perspective the clinical laboratory is composed of different departments with particular areas of expertise, such as Microbiology/Immunology, Haematology and Biochemistry. The ASK-agent architecture includes the *ownership* attribute within the service description to manage this aspect of the laboratory structure. But guidelines themselves do not contain any evidence that identifies rules which relate to an inter-departmental standpoint. This means that if the ASK-agents are required to be divided up into departments then a separate set of management rules needs to be developed. The formation of expert groups is present within the guideline documents because they focus on the disease, condition or organ from an expert opinion perspective, and use a particular ontology in which to disseminate their understanding of the domain.

Finally the contribution to the body of knowledge the ASK-agent architecture makes was discussed in section 7.4. It showed this novel approach can preserve the original guideline motivation within a standalone entity, permitting the ASK-agent to have autonomous control over its activities, without the requirement of developing a set of management rules. It showed that the ASK-agent architecture permits the individual agents to have representation, awareness, investigation, interaction, integration and coordination with other ASK-agents. These were challenges Corkill defined as critical in the formation of collaborative groups of autonomous software modules. This research illustrated that the proposed architecture allows all agents to have awareness and representation of the services each other ASK-agent can provide. The implementation has also demonstrated how information can be exchanged in order to utilise those services. Therefore there is no barrier, such as having to channel all queries through a centralised management resource, for other agents to access the motivation, knowledge and logic the individual ASK-agent contains. The implementation demonstrated that the ASK-agent validation reports not only show the result under scrutiny, but also presented a list of comments other ASK-agents in the system made during the validation. It is not necessary that this information is forwarded to the clinician, but if it was it could be utilised to support their patient care activities, or identifying sample result irregularities that may have gone unnoticed during routine testing. As the ASK-agent is the only custodian of the guideline motivation, knowledge and logic it can be deleted, altered or added without any other module requiring alteration. As discussed in section 2.4.2, the majority of laboratory tests have an Index of Individuality <1.4 which is a limitation if they are to be used for screening or diagnosis. The proposed ASK-agent advances the II for laboratory tests by (a) stratification using searchable expert ASK-agents, and (b) communication between groups of expert ASK-agents.

Chapter

8

Conclusion and recommendations

8.1 Introduction

This final chapter is divided into four sections. Section 8.2 reflects on the underlying requirement for patient-centred validation. It focuses on how autonomous intelligent agents can realise it from both an intra-module and inter-module standpoint. Section 8.3, presents the conclusions of this thesis, and finally section 8.4 ends this chapter by discussing various potential avenues for future research.

8.2 Research overview

This research has described laboratory results as an indicator or "*snapshot*" of patient health. The clinicians use results to aid them in making a diagnosis or manage patient healthcare with a greater degree of certainty. This supportive information is only of use to the clinician if it is accurate and plausible for the specific patient from whom the sample was taken. It is the clinical laboratory's responsibility to validate the authenticity of these results before issuing them to the clinician. Although the overall validation cycle was shown to be complex and multifaceted, the fundamental issue being addressed is the method by which the validation is performed. The following sections describe some of the multifaceted aspects associated with laboratory validation. In so doing, these sections progressively introduce design aspects affecting the structure of the ASK-agent architecture, which has been developed as part of this research. Then finally this section discusses the laboratory setting in which the ASK-agent was evaluated.

Validation Cycle

When a clinical laboratory test is ordered for a patient, the sample is taken, and then the sample is sent to the clinical laboratory for analysis. The laboratory analyses the sample and produces a set of analyte results. An analyte of a sample result is the constituent of the specimen to be measured. But before the result can be reported to the clinician, the laboratory staff must ensure the results are valid and plausible for the specific patient from whom the sample was taken. This is a critical link in the overall validation cycle because these results are used as a decision-making aid by the clinician. An error in the accuracy or plausibility of this result could therefore have a detrimental effect on the patient's health. After examining the overall validation cycle from the request through to reporting (see section 2.3), it was established that the key influencing factor affecting the authenticity of results for a specific patient related to the management of biological variation. Biological variation refers to changes in a patient's analyte result which is based on gender/age, biological cycle rhythm, and presence/absence of disease/illness. This was because the outcomes from the other factors of the validation service do not use, and are largely unaffected by patient specific data. It was also shown that the index of individuality (II) of the tests also affected validation (see section 2.4.2). The II of a laboratory test is an indication of how results of a test relate to a group of patients. If a set of samples from a group of patients was analysed using a specific laboratory test, and all the result variation fell neatly into an narrow range then the test would have a high II i.e. test has little individuality between patients. Tests with a high II >1.4 are good for screening and diagnosis. However, if the test has a low II <0.6 then the result variation would be scattered, making it impossible to set upper and lower reference ranges. Tests with a low II <0.6 are useless for screening and diagnosis. However by stratifying the patients into different categories based on gender, age or ethnicity or combining the analyte result from different tests the II for a test can be improved. This research highlighted that most tests in the clinical laboratory have a low II (<0.6). This means that more patient information or additional analyte data are required to accurately validate results for a particular patient.

The existing Computer-Interpreted-Guideline approach used by laboratories

The existing laboratory approach to patient result validation is based on the Computer-Interpreted-Guidelines (CIG) model. Using this model the contents of narrative clinical and laboratory guidelines are decomposed into computerised workflow components and management rules. The separate workflow components describe particular procedural paths based on accepted best practice information expressed within clinical and laboratory guidelines. They use patient specific characteristics at key junctures to alter the flow. The separate workflow components are described using Arden Syntax Medical Language Modules (MLM) (see section 2.6). The Arden Syntax is a rule-based formalism developed for encoding individual clinical rules as MLMs. Each MLM is written to behave like a single rule or activity. To realise this each MLM uses four main "slots": an "evoke slot", a "data slot", a "logic slot" and an "action slot". The evoke slot acts as the "trigger" (e.g. upon admission of the patient). The data slot as a retriever of the relevant data from the LIS (i.e. PatientWeight). The logic slot uses this data in an if-then-else type arrangement, where the "IF" part of the logic statement is the logic test (e.g. PatientWeight>= 80kg), and then the action slot executes the "then" piece of the statement. The CIG centralised engine manages MLM selection through a set of management rules.

This approach is successful to a degree in validating patient laboratory results, but the operation is essentially validation-centred, and not patient-centred. The approach validates the results in accordance with a specific workflow procedure, but does not make use of all available supportive information from the respective guidelines. When a new CIG workflow MLM is constructed from a guideline, the management rules in the CIG centralised engine must be altered. Only after the additional management rules have been applied can the new workflow element be accessed and utilised. This reconfiguration overhead requires a new CIG engine to be compiled from the source code. This revised CIG engine must be installed resulting in system downtime. Once a new workflow element is included within the main collection of CIG's, the original guideline motivation, knowledge and logic is no longer self-contained and cannot be independently accessed. It is therefore an integrated part of the engine.

Patient centred validation

When a clinical or laboratory guideline is developed by an expert group they focus on best practice for the specific disease, condition or organ. They include all relevant knowledge, logic and motivational aspects they deem necessary to adequately describe the domain. Therefore, the guideline would be considered by the authors as a complete and self-contained body of knowledge in that specific domain. This means that the guideline information should be viewed holistically, and not just as a collection of separate workflow elements taken out of context. True patient-centred validation is where the guideline is viewed and interpreted as a whole to make use of the maximum supportive information based on patient specific characteristics. The novel approach provided by using autonomous socialising knowledge agents (ASK-agent) developed in this thesis addresses this problem.

Agent oriented approach

The agent-oriented approach operates on the notion that human intelligence and decisions can be synthesised by managing tangible elements, such as beliefs, goals, sub-goals, plans and intentions. The agent-oriented paradigm advances the Object-Oriented (OO) paradigm by adding a formalised description for motivation in the form of goals, rather than dictated by its location in the computer code. Agents are then programmed using belief, plan and goal attributes which are managed by the agent's inference engine. The agent's inference engine searches for plans (actions) to execute based on chosen goals (motivation) and known beliefs (facts). The selection of an agent's plans and goals are based on preconditions, conditions or triggers which are fulfilled using beliefs and logic (see section 3.2 and 3.4). The relationship between beliefs, plans and goals to produce the desired action has been the subject of debate for a number of years. As a result various base-level agency platforms have been developed which experiment with different combinations of logic in which to manage these elements. A base-level agent platform provides generalised tools for describing beliefs, goals and plans and includes an interpretation engine. One such interpretation is known as Beliefs, Desires and Intentions (BDI) although others exist (see section 3.4). The rationale for choosing BDI was the similarity between BDI and guideline characteristics as shown in Table 8.1.

BDI Structure		**Characteristics**		**Guideline Structure**
Belief	↔	Informational attributes	↔	Facts
Desire	↔	Motivational attributes	↔	Goals and aims
Intension	↔	Deliberative attributes	↔	Selection of Actions

Table 8.1 Comparison of BDI to Guideline characteristics

Within the group of BDI based agents further refinements on the inference engine have been realised through agency platforms such as Jason, Jadex, 3APL and PRS. These are generic base-level agency shells that can be adapted by the users for a large variety of different applications. Using the domain specific characteristics a review of agency based platforms was undertaken. After comparing the domain and agent characteristics, Jadex was chosen as the most appropriate base-level agent on which to build the ASK-agent application (see section 4.8).

Generic guideline composition

Although there are characteristic similarities between the base-level agent and the guideline, Jadex is only a shell and does not possess the ability to operate on behalf of a guideline without significant additional functionality. To this end, guideline composition was examined to identify reoccurring structures, operations and themes. These aspects are used to describe guideline knowledge and workflow activities, such as decision steps, action steps, synchronisation steps and decision priority. After examining a range of different guidelines in various medical disciplines related to this research domain, a number of these reoccurring aspects were identified and discussed (see section 4.2). For all aspects identified in this section, an agent equivalent component was developed and added onto the base-level agent as detailed in Appendix C. To add further focus to the capturing of guideline knowledge, logic and motivation, the Arden Syntax Medical Logic Module (MLM) approach was expanded in Section 5.33. This expansion took the form of adding achieve goals, maintenance goals, query goals, meta-level reasoning, modal reasoning, language, ontology and service descriptions to the existing MLM slots. The meaning and operation of these components is described in Table 5.5. This effectively added onto the existing Jadex agent shell the necessary components for an ASK-agent to act using BDI captured from a guideline. It is important to emphasise the types of guidelines being discussed are text based documents with conclusions being derived using numerical analysis as illustrated in all the examples used in this thesis. Neither the proposed ASK-agent nor existing CIG approaches can manage guidelines which have been developed using graphical imagery and charts.

Retrieving data from Laboratory Information System

A Laboratory Information System (LIS) is a software application, or more specifically a group of software applications, which are used by laboratory staff to manage and coordinate their day to day activities. The general functions they perform are the storage of data (e.g. patient results, gender, age, height), report generation and also provide a mechanism to interpret sample results and manage process workflows. Section 2.6 of this thesis examined the operations of the LIS and described it as a system which essentially provides three key processes:

1. A method of describing and representing the context of data, so it can be accurately and correctly interpreted.

2. A method of communicating the data in a manner, which preserves its context as described in (1) between modules.

3. A method of representing and executing logic and/or workflow, which can be executed by the system's interpretation/inference engine.

It is common for these three generic processes to be repeated in various forms throughout the LIS. Examples of the data context are where patient data is stored in the analyser, in the database and in the reports generated. Examples of communication are from the analyser to the LIS, from the LIS to the clinician and from module to module within the LIS. Finally examples of the logic and workflow are inference engine, Standard Operation Procedure (SOP) work-list and workload planner etc. Although guideline-based decision support systems and the ASK-agent primarily focus on the interpretation/inference engine resource, the method of describing the context of data and its communication between separate modules must also be examined. This is because the interpretation/inference engine resource is predominately dependent on the other two facilities.

To this end it is necessary for the ASK-agent to be able to access patient data stored in the LIS. It is possible for the ASK-agent to execute any Java plan. Therefore, an ASK-agent can perform a Structured Query Language (SQL) query on the LIS database to retrieve the necessary data to perform the validation. However, the SQL language just retrieves the actual data not the structure or context. In the revised CEN pre-standard prEN13606:2004(E) which is currently under construction, an archetype is being specified which simplifies the retrieval of data whilst using distributed systems (see section 3.5). An archetype is a re-usable, template model which presents a specific viewpoint of a domain reference model. Using the archetype approach both data and context are preserved. The ASK-agent architecture permits the development of an archetype query to be structured and stores it in an Archetype Repository (AR) agent. The ASK-agent sends a message to the AR agent to perform the query. The AR agent performs the query and returns a message containing the requested data in the form of an XML, Schema or Object. The ASK-agent can then use the information in order to validate the patient's sample.

Communication between Agents

Indeed the fundamental power behind the agency approach is the ability of each agent to be self-governing and autonomous. However, to truly reproduce the way in which guidelines are used, a mechanism must be provided to allow loosely coupled ASK-agents to cooperate, collaborate, and negotiate to achieve common goals. This social aspect is one of the main characteristics of the agency paradigm, where agents socialise and work together in groups. Therefore it is vital the communications between agents and other users are carefully constructed and controlled.

From a medical perspective message transmissions between separate software modules must comply with a medical standard for communication, such as CEN ENV 13606-4:1999 (see sections 3.5 and 4.4). This is to ensure the message arrives at the correct location and the context/meaning of the message is preserved. The standard achieves this through two components: (a) message payload, and (b) message envelope. The payload focuses on the message context and meaning. A payload can be an XML document, Schema or Object. The envelope identifies and traces the messages progression during delivery. The envelope is a list of parameters such as sender ID, priority level and thread ID etc. From the agency standpoint the most widely used communications standard is that produced by the Foundation for Intelligent Physical Agents (FIPA), although other standards exist. The FIPA communication standard structures messages in a similar way to that provided in the CEN ENV 13606-4:1999 (i.e. using payload and message). But the main difference is that agents use messages for both information passing and social interaction. Sections 3.5 and 4.4 described and analysed both the medical CEN ENV 13606 standard and FIPA's standard for communications. It showed that six of the twelve CEN ENV 13606 parameters have similar meanings with the FIPA standard. It also showed that the FIPA standard could be adapted to include the missing parameters so it could comply with the medical standard, yet retain its social capability.

Ontology and language

Difficulties in communicating and sharing medical information between institutes, individuals or groups have generated a multitude of ontology and language implementations. Examples of medical ontologies are Galen, Tambis, UMLS, ONIONS, HL7 RIM and GENE. This demonstrates that no unique "one-stop-shop" ontology for the medical domain exists. This is because an ontology is only a formalised description of the real-world environment. The FIPA agent message structure recognises that in the real-world different ontologies exist. Therefore instead of forcing a single ontology it allows many to exist in the same environment and includes a framework to define, describe and manage them. FIPA's agent ontology and language framework is a richer structure for expressing context and meaning than the archetype described above. It can be used to improve the inter-ASK-agent communication to a level beyond that used by archetypes.

Collaboration between Agents

Now that the ASK-agents architecture permits an agent to work autonomously and communicate with other agents, the focus changes to how these separate entities can work together or collaborate on activities. From a medical perspective this is where two or more guidelines could be applied to a single patient. But the intra-agent BDI or the communication standard does not explicitly detail how the agents can socialise, collaborate or form groups. Therefore, the relationship interfacing between separate ASK-agents needs to be addressed.

As a consequence of the agent architecture simulating human decision-making and reasoning, albeit at a high level of abstraction of real human reasoning, agent sociality can be compared to human sociality. From an inter-agent social and interactive perspective there are a number of theories which have been developed through the social sciences (see section 3.8). These theories focus on the collaborative nature of separate autonomous systems on which agents are based and have the capability to complete certain tasks jointly. For an ASK-agent to truly collaborate with another, it needs to be able to pursue activities that may not lead to the goal of result validation, but through its action support the validation of the result by other agents. For example, an agent requiring a Creatinine analyte result could use a blood based Creatinine result or urine based Creatinine result in order to progress a validation task. This would permit the validation procedure to continue, using this additional knowledge artefact. After discussing and examining alternative approaches the Activity Theory was chosen and used to study this validation domain (see section 4.6). When applying this theory to interacting clinical and laboratory guidelines it illustrated that although the guidelines appear to be standalone documents, groups of them are coupled together by virtue of their overlapping knowledge. The overlapping knowledge is provided in two main forms. The first is in the form of similar domain knowledge that uses alternative inference mechanisms in order to derive a result (i.e. both statistical and rule-based inference engines being able to validate the same result). The second is in the form of overlapping knowledge which observes different viewpoints of the same domain. For example, the kidney filters toxins from the blood passing it to the urinary tract. As the kidney is such an integrated organ in the body there are many guidelines describing its operation from different viewpoints such as blood filtering, urinary tract, autoimmune disorders etc. Using this approach the organ disease or condition is described from different viewpoints through various guidelines. Each guideline describes different knowledge, logic and motivational aspects associated with the organ. Therefore supportive information can be exchanged between these guidelines in order to aid in describing the operation of the organ, or in the validation of a sample result. To utilise this overlapping knowledge link a social structure was developed to manage the interfacing between agents. This social interface took the form of a mandatory set of searchable service descriptions, beliefs and actions. The service descriptions (i.e. *Name, Type, Ownership, GuidelineReference, InformationNeeded, ValidationType, EndResultType, Ontology and Language*) permitted each agent to be located within the agency platform through the Directory Facilitator (DF) (a feature of the FIPA standard offering searchable *golden-pages* facility to locate agents) (See section 4.9.1). The beliefs (i.e. CurrentlyValidating, PlausibilityScore and localised blackboard) permit the ASK-agent to interact with other group members (See section 4.9.2). The actions relate to automated responses the ASK-agent must return to other agents when queried (e.g. CurrentlyValidating), and the sending of information to other agents it believes should be reported (e.g. it determined the presence of liver disease during its deliberation). Therefore each ASK-agent only needs to know its overlapping neighbours, which it can find and interact with using the agent platforms DF and message passing. With access to supportive and overlapping knowledge it is not necessary to have a single all-encompassing rule set to manage the ASK-agents' interaction.

Autonomous socialising knowledge agents (ASK-agent)

The novel ASK-agent architecture was developed by combining functional aspects of existing software and communication systems and adding new components and techniques to address aspects of the domain under investigation. By combining these components the ASK-agent has the ability to act intelligently on behalf of the clinical or laboratory guideline. In addition to this the ASK-agent retains a social interface to allow it be contacted and used by others to support patient-centred validation. Therefore, this approach utilises both intra-guideline aspects which are used to process the individual ASK-agents internal course of action, and the inter-guideline aspects which are used to manage the ASK-agent-to-ASK-agent relationships. Using this approach a validation service can locate and communicate directly with the ASK-agent, accessing the maximum supportive information which the module can offer from the full depth of its knowledge about the particular domain and the individual patient. This operation dynamic is fundamentally different to the restrictive workflow validation cycle provided by the CIG approach. As a result it provides a more comprehensive form of validation which is customised according to specific patient data.

Medical domain

In order to ensure that the ASK-agent system can accurately represent clinical and laboratory guidelines and protocols it is important to choose a domain that can test a validation prototype. Therefore, the choice of medical domain in which to demonstrate the ASK-agent approach must be carefully selected to emphasise the generic, yet different, aspects of result validation (see section 2.7).

The main criteria for selecting an appropriate medical domain was:

1. Analyte results should be from a well established group of tests or procedures.

2. Analyte should be from tests that use control samples which produce a narrower variation in the reported results and can be validated using numerical examination.

3. The validation procedure should support multi-validation paths which can be used to dynamically choose an appropriate route to validate the results based on known patient information. This allows the ASK-agent act autonomously.

The search for an appropriate medical domain focused on three main organs of the body, namely Heart, Kidneys and Liver. The medical domain focusing on the Heart is limited, because the principle methods of assessing the Heart are ultrasound, x-ray, resonance scanning etc. Outcomes using these methods are not always based on numerical analysis, and therefore are not suitable for using CIG and ASK-agent approaches. Results used in the Kidney domain are currently available in the laboratory without any alteration to work practices, and are mostly validated using numerical analysis. The domain is based on different interfaces with components of the body specifically blood, urine and bowel movement. Therefore, the Kidneys are a very appropriate domain for an agent validation system. But an even stronger medical area based on the selection criteria is the Liver. The laboratory currently generates results used to evaluate the Liver based on pure numerical analysis. The Liver interfaces with the blood, urine and digestive system, and its validation techniques can be separated into autonomous systems. The domain is well documented and accepted definitions for specific analyte results are available and written into guidelines. Within the domain of the liver there is an approach were Liver Function Test (LFT) results are used by clinicians as a key indicator to the source of anaemia for chronic diseases. Anaemia is the most common disorder of the blood and is described as a deficiency of red blood cells and/or haemoglobin in the blood. In most cases it is indicative of a chronic illness or cancer, but in certain cases anaemia is normal due to the metabolism of the patient.

LFTs are a suite of tests applied to serum samples to determine the level of key analytes in the blood. These analytes establish the status of liver membranes which are used to indicate its level of operation. The Index of Individuality score for each of these tests is

<0.99 with some being <0.4. This makes them unsuitable for diagnosis or screening unless some form of stratification takes place. If certain parts of the liver are damaged then the LFT will indicate substantial changes in analyte results. The analytes reported as part of the Liver Function Test in particular the serum '*Total Protein*', '*Albumin*', '*Total Bilirubin*', '*ALT (Alkaline aminotransferase)*', '*Alkaline Phosphates*' and '*GGT (Gamma-Glutamyl Transpeptidase)*' are key indicators which can be used to categorise the source of different types of chronic anaemia. Examples of chronic anaemia assessed in this way are Kidney disease, Liver disease, Alcohol disease and Inflammatory disease. This is because there is a correlation between types of diseases and illness associated with combinations of analyte results. This overlapping of knowledge using combinations of analyte results improves the II of a set of tests, making them accurate indicators for screening and diagnosis.

8.3 Conclusions

The main goal of this thesis was to investigate patient-centred clinical validation using autonomous socialising knowledge agents (ASK-agent). The thesis proposed that the ASK-agent architecture provides a novel opportunity for patient-centred validation to occur using just guidelines without the need for a centralised inference engine. The development of the system was completed strictly in accordance with the procedures described in chapter 5. Each module was tested individually using test data to ensure it operated correctly and represented the guidelines content, thereby verifying the intra-ASK-agent operation. The four agents were then run concurrently and tested using additional data sets. This testing verified the inter-ASK-agent operation, and ensured that they interpreted incoming messages correctly, and issued supportive messages to other members in the group. It was only at this stage the operation of the four agents was evaluated using actual patient data, taken directly from documented medical case profiles with known outcomes.

The results returned by the ASK-agent were independently verified by comparing them to outcomes produced by a human expert who reviewed the data given to the ASK-agent, and the factual medical conclusion provided in the case history used. This showed that the ASK-agent outcomes were the same as the human expert and the factual medical conclusion. The implementation therefore demonstrated that guideline motivation, knowledge and logic are presented in a format that can be coded within a standalone ASK-agent, and when incorporated using the novel ASK-agent architecture they can interact to validate patient samples.

However, there is an additional dimension in relation to guideline selection that needs to be discussed. For a group of ASK-agents to interact, socialise and collaborate on shared goals and activities the chosen guidelines must have a degree of overlapping knowledge as discussed above. Without this overlapping knowledge the ASK-agents are technically searchable islands of information with little or no opportunity for collaboration to take place.

Therefore, if this approach was to be considered for development on a green field site it is important that the hospital's medical disciplines (which the laboratory supports) are not standalone. Different departments must be in a position to collaborate together to validate results in a truly patient centred fashion. This would dramatically support the underlying operation of the ASK-agent approach. Once this overlap is present the narrative based guidelines can be converted into ASK-agents, and the system can be developed and tested in a similar way to that used and described above for the implementation.

During the development of the ASK-agent architecture implementation, a number of performance related issues were encountered which could affect its scalability and compatibility with existing laboratory systems. As the ASK-agents become increasingly distributed onto separate computer systems within the laboratory there will be a marked increase in the volume of network traffic. This is due to the ASK-agents reliance on message passing in order to exchange information, and the size of the message required to adequately describe its contents and context. However, this network traffic overhead can be dramatically reduced by selectively grouping the ASK-agents and managing their interactions. The developed architecture is based on ASK-agent modules being autonomous self-contained software entities that can be searched using the DF which can access the guideline knowledge, logic and motivation in its entirety. This approach is fundamentally different to the existing CIG based approaches and can affect its compatibility with existing laboratory systems. Therefore, if legacy laboratory systems were to be reused, access to them would need to be wrapped using an additional software middle layer to include the missing motivational components which were removed during the decomposition of the original guideline.

A summary of the main differences between the CIG and ASK-agent approaches is given in Table 8.2.

Element	CIG	ASK-Agent
Multi-ontologies	Not able to manage multi-ontologies. See section 4.3.	Can manage and operate using multi-ontologies. The ASK-agent base-level platform is Jadex which permits and manages multi-ontologies and languages. But the use of multi-ontologies adds an additional complication to the execution.
Organisational structure	Centralised. Knowledge sources can only be accessed via the main CIG engine. No independent access is allowed.	Distributed. All ASK-agents can operate and be accessed independently using message passing.
Size of file(s) created	For a single guideline the CIG engine is small, because only guideline extracts are used. However, as more MLM's are added the CIG engine application file becomes very large. Processing cannot be distributed.	For a single guideline the ASK-agent is larger than the CIG, because a lot more of the guideline is encoded. However, as the model is distributed the system file size is not a limitation.
Requirement for overlapping knowledge between guidelines	The CIG approach does not need overlapping to operate. The CIG engine is a centralised source of management rules which manage the knowledge. Information only has to relate to the workflow dictated by the management rules.	The ASK-agent application is dependent on overlapping knowledge to exist within the guidelines. As no centralised management rules exist the ASK-agent is reliant on the sharing of knowledge from different view points in order to establish links.
Addition, altering or removal of guidelines from the system	Any changes to the CIG system such as adding, altering or removal, requires for the management rules in the centralised engine to be recompiled. This requires considerable down time and reinstallation of the software.	As each ASK-agent is independent and loaded separately on to the base-level platform. It is not necessary for the system to stop completely. In most cases there will be a number of alternative paths which could be used to validate the samples.
System resilience	If the CIG engine crashes the application stops operating. No system resilience included.	If an ASK-agent crashes, another instance of the same ASK-agent already loaded on the platform can be used. The blackboard from an existing ASK-agent in the group can be copied over to the new ASK-agent, thereby providing system resilience.
Independent accessible knowledge	The CIG application does not allow for the separate workflow paths or knowledge sources to be accessed by anything other than the centralised engine.	The ASK-agents contain the information in a format which is totally independently accessed, described in the Directory Facility DF *golden-pages* and available to all parties via message passing.
Network access to retrieve data	The CIG application accesses data from healthcare record or Laboratory Information System (LIS) via the network.	The ASK-agent application accesses data from healthcare record or Laboratory Information System (LIS) via the network.
Network access to execute logic	The CIG application does not need to use the network in order to process logic. All logic is executed within the centralised engine.	The ASK-agent application needs and is reliant on the network in order to communicate information to other ASK-agents via message passing. Although a single ASK-agent processes its intra-logic itself without the need to access the network, it has a duty to forward information it believes would be of use to others. Groups of ASK-agents can be located on a single machine to minimise network traffic.
Clinician having access to specific guideline knowledge	The CIG application only uses guideline extracts. A clinician cannot access the knowledge directly.	The ASK-agent application uses message passing. A clinician can access each ASK-agent via a message and directly access the specific guideline knowledge.
Access to the LIS	The LIS is an integral part of the CIG systems operation. Results are sent to the CIG from the LIS for validation. The CIG has access to slots in the LIS to retrieve data to aid in its execution. The CIG reports its findings back into the LIS.	The LIS is part of the ASK-agent systems operation. Results are sent to the agent manager by the LIS. At the end of the deliberation the agent manager collects the separate reports and deposits them in the LIS. An ASK-agents can order the execution of archetype queries of the LIS and other data sources through the AR agent.
Corkhill's challenges for collaborative software	GIG's only address three of Corkhill's challenges for collaborative software.	ASK-agent addresses all six of Corkhill's challenges for collaborative software.
Method of collaboration	Direct links using the CIG engine based on the centralised blackboard as no knowledge source can be identified or accessed independently.	Using the mandatory beliefs, action and descriptions in the agent platforms Directory Facilitator, each knowledge source can be located and its knowledge accessed independently.

Table 8.2 Summary comparison of CIG and ASK-agent approaches

In achieving the six aims as stated in chapter 1 this thesis makes the following contributions:

1. It created an innovative approach to validating patient specific data using distributed resources, without the need for a centralised element of control to manage the interfaces between these resources. This was accomplished by having the ASK-agents act autonomously and replicating information using the local blackboard. It is important to emphasise that the purpose of the manager agent was purely to act as a gateway between the LIS and the ASK-agent architecture. It is only used to start the validation when a set of patient data is sent to it from the LIS, and to return the report at the end of the validation deliberation. Many of these manager agents can exist on the system concurrently and they do not manage or take part in the actual deliberation activity itself. The deliberation activity is the responsibility of the group of ASK-agents.

2. It advanced agent architecture by improving their ability to collaborate by introducing three new customised components. These generic components introduced as part of the ASK-agent template provided a social fabric, so separate agents collaborated on related activities.

3. It formulated a guideline-to-ASK-agent map which interpreted and transformed the intention, knowledge, logic, goals, meta-level and modal reasoning aspects of guidelines into the ASK-agent module components.

4. It constructed a system that allows an ASK-agent to be added, deleted or altered without the need to reconfigure the system. This was achieved by eliminating the need for a centralised control, and retaining control as a distributed component as part of the standalone ASK-agent. In addition to this the presence of the local blackboard in each ASK-agent can offer a form of system resilience in the event of an ASK-agent failure, or when a new ASK-agent is called upon to provide supportive information. An existing ASK-agent can replicate the currently held blackboard information to the new or

replacement ASK-agent. This would inform the new ASK-agent of the current state of deliberation among the group, and avoids the need to restart the validation cycle again.

5. It provided a deeper understanding of how information, knowledge and logic are described in the clinical and laboratory guideline documents. This provides a greater insight into ways in which these aspects could be better described and constructed by authors of future guideline documents.

Although the validation application used for the ASK-agent was based on the human medical domain, the architecture itself could be applied to other domains. Consider for example a completely different application, that of maintenance faultfinding for an automated building air-conditioning process or chemical process. Each plant component whether it is a refrigeration gas compressor, water purification system, chemical dosing, or pH control has its own set of guidelines for operation and maintenance. Each guideline focuses on individual plant components associated with the process from different expert group perspectives, say of the mechanical, electrical, control or process engineers. These guidelines present all relevant knowledge, logic and motivational aspects the experts deem necessary to adequately describe the domain. An analogy of this could be the industrial plant being the human body and the individual plant components being the heart, kidneys and liver, and the various types of engineers being the different clinician disciplines. All the guidelines provide overlapping knowledge of the plant components from the respective expert's viewpoint, with a view to maintaining the operation of the process. In the majority of cases the control system which manages this complex process would be developed using programmable logic controllers (PLC's), and Supervisory Control and Data Acquisition (SCADA) systems. This means that the knowledge, logic, and motivation within the guidelines would already have been adapted for use on a form of computer system and has a foundation on numerical based knowledge and logic. The presence of the two overlapping knowledge forms described above is present (a) similar domain knowledge using alternative inference mechanisms (e.g. rule-based maintenance and statistically-based fault occurrence knowledge), and/or (b) overlapping domain knowledge focusing on observing different viewpoints of the same domain through the different engineering disciplines. These domain characteristics match the required features of the ASK-agent architecture making this application an ideal candidate for using the developed ASK-agent approach.

8.4 Future work

There are various potential avenues of research relating to the work presented in this thesis that may be investigated. Some of these avenues are discussed below:

Social interaction

This thesis has identified that agent orientated programming is based on the societal view of computation, but the social aspect of existing agent systems is performed on an ad-hoc basis. The social fabric aspect is tremendously important in developing the ASK-agent, because the ASK-agents are designed to act autonomously and must socialise and communicate with each other in order to work in groups and collaborate. The structure of the different types of social activity, for example broadcasting, lecturing, meetings, interacting and collaborating, are dramatically different and further research in this area would improve the system's social interaction efficiency.

Customised functions and rules

Existing CIG systems provide a selection of dropdown menus and templates where the reoccurring structures, themes and operations in the guideline can be quickly encoded by the programmer. The ASK-agent system still requires the designer to develop the guideline using the ADF XML file and the plans in Java. These are files used to encode an ASK-agents Beliefs Desires and Intentions. Further work could be completed in automating the ASK-agent versions of these reoccurring features developed in Appendix C and the extended MLM as developed in 5.3.3.

Service description parameters

This thesis has demonstrated that by creatively structuring the content of the *Name, Type, Ownership, GuidelineReference, ValidationType, InformationNeeded, Ontology* and *Language* parameters of the agent service description, a host of specific medical services can be explicitly declared to reflect the true operation of the ASK-agent. Further research into specifying specific medical services using these parameters could be undertaken, to improve the systems efficiency.

Inter-department communication

The departmental structure which exists within clinical laboratories as discussed in section 2.7, depends on the hospital core activities, and political/financial pressures. Therefore the range of analysis and services each department provides can change over time, leading to an increasing number of cases of test duplication between departments occurring. For example Red Blood Cell counts are performed simultaneously in the haematology, biochemistry and biology departments of the clinical laboratory, on the same patient sample. The guideline documents are not developed from a departmental perspective, but more from a disease, condition or organ focus, and so any correlation to a particular department are inferred rather than explicitly stated. Patient-centred validation does not recognise the presence of departmental borders. However, from a pragmatic perspective where laboratory departmental structures are in existence these borders can be realised using the *Ownership* attribute in the service description. It is not necessary for this parameter to be used from the ASK-agent perspective. But where departmental borders exist from an institutional standpoint this constrain can be utilised to provide a form of departmental groupage. Further research into managing this activity in order to make the ASK-agent architecture more suitable for established laboratories settings would need to be undertaken.

References

3APL (2005). http://www.cs.uu.nl/3apl/, accessed June 2005.

ADA (2006). American Diabetes Association (ADA). "Standards of medical care in *diabetes, II. Screening for diabetes"*. Diabetes Care January 2006; 29 (Suppl 1):S5-7.

AGA (2002). "*Guideline-Medical position statement: evaluation of liver chemistry tests*". Gastroenterology published by American Gastroenterological Association 2002, Volume 123 Issue 4, pages 1364-66, accessed via http://www.gastrojournal.org/home.

AGAM (2002). American Gastroenterological Association. Medical position statement: evaluation of liver chemistry tests. Gastroenterology 2002, 123, pages 1364-1366.

AgentLink (2005). http://www.agentlink.org/, accessed May 2005.

Aglets (2005). http://sourceforge.net/projects/aglets/, accessed May 2005.

ASCLS (2005). http://www.ascls.org, accessed May 2005.

AUML (2005) http://www.auml.org/auml/main.shtml, accessed January 2005.

Bachmann C., (2000) *"The control of analytical accuracy and day to day precision helps for the follow-up of patients and is essential when using biochemical data from the literature"*, published as European Research Network Inherited disorders of metabolism (ERNDIM) Biomed2 Newsletter Centre Hospitalier Universitaire Vaudois Number 3, May 2000. Accessed via http://www.erndim.unibas.ch/pdf/control.pdf

Baker Patricia G., Brass Andy, Bechhofer Sean, Goble Carole, Paton Norman, Stevens Robert, (1999). *"TAMBIS-Transparent Access to Multiple Bioinformatics Information Sources"*, Scientific and Statistical Database Management, 1999. Eleventh International Conference on 28-30 July 1999 Page(s):280

Beeler George, W., (2001) "*Reference Information Model (RIM)*" HL7 Version 3 tutorial in two parts Part I & Part II accessed via http://www.hl7.org, 2001.

Benson Tim, (1998) "*Short Strategic Study: Enabling Technologies-UML (Final Report), CEN/TC 251/N98-082*" published by EUROPEAN COMMITTEE FOR STANDARDIZATION COMITÉ EUROPÉEN DE NORMALISATION EUROPÄISCHES KOMITEE FÜR NORMUNG Version 1.21, 25 May 1998

Blackburn Patrick, de Rijke Maarten, Venema Yde, (2001). "*Modal Logic (Cambridge Tracts in Theoretical Computer Science)*" published by Cambridge University Press, June 2001, ISBN-10: 0521802008.

Boran, G., Given, P., O'Moore, R., (1996) "*Patient result validation services*", Computer Methods and Programs in Biomedicine Vol 50 (1996, Elsevier Science Ireland Ltd) pages 161-168.

Bordini ,Rafael H. Hubner Jomi F., (2005) "*Jason: A Java-based agentSpeak interpreter used with saci for multi-agent distribution over the net*" accessed via the web from Department of Computer Science, University of Durham, Durham DH1 3LE, U.K. March 2005

Bratman M.,(1987). "*Intention, plans and practical reason*", Harvard University Press, Distributed for the Center for the Study of Language and Information, 1987, ISBN:9781575861920.

Braubach L., Pokahr A., (2005). "*Jadex Tutorial*", provided as part of the Jadex platform documentation by Distributed Systems Group, University of Hamburg, Germany via http://vsis-www.informatik.uni-hamburg.de/ Release 0.932, 10 May 2005.

Braubach L., Pokahr A., Krempels K., Lamersdorf W., (2004). "*Deployment of Distributed Multi-Agent Systems*" Fifth International Workshop on Engineering Societies in the Agents World (ESAW-2004).

Brown JB, Shye D, McFarland B, (1995). "*The paradox of guideline implementation: how AHCPR's depression guideline was adapted at Kaiser Permanente Northwest Region*", The Joint Commission journal on quality improvement, 1995;21(1):5–21.

Browne Eric Donald, (2005). "*Workflow Modeling of Coordinated Inter-Health-Provider Care Plans*", School of Computer and Information Science, University of South Australia, Adelaide, PhD Thesis January 2005

Burtis Carl A., Ashwood Edward R., Bruns David, (1998). "*Tietz Textbook of clinical chemistry*", published by Saunders W.B, ISBN 0-7216-5610-2, published in 1998, accessed via short term loan DIT Kevin Street.

Bury G, O'Dowd T, (2006). The Irish Medical Council, "*Internal Job Description and Log Book*", accessed via http://www.medicalcouncil.ie, accessed June 2006

Buttner J, Borth R, Boutwell JH, Broughton PMG, (1976). "*International Federation of Clinical Chemistry provisional recommendation on quality control in clinical chemistry. I. General principles and terminology*". American Association for Clinical Chemistry Journal 1976;22:532-40.

Cabot Sandra (2004) "*Tests for Liver Function*", accessed via http://www.liverdoctor.com/Section2/09_livertest.asp on April 2004

CAP (2005). *Today "LIS-survey" College of American Pathologists*, CAP Today Magazine, November 2005, pages 24-56, http://www.cap.org/apps/docs/cap_today/surveys/11_05_24-56_LISsurvey.pdf.

Carey William (2006). "*A guide to commonly used liver tests*" accessed via the web at http://www.clevelandclinicmeded.com/diseasemanagement/gastro/livertests/livertests.htm, July 2006.

Carey William D., Fise Thomas F. (1993). "*Gastroenterology Practice Management*", published by Igaku-Shoin Medical Pub, May 1993, ISBN-10: 0896402371.

Casiday Rachel, Regina Frey, (2001). "*Maintaining the Body's Chemistry: Dialysis in the Kidneys*" accessed via web http://www.chemistry.wustl.edu/~edudev/LabTutorials/Dialysis/Kidneys.html, revised January 2001

CEN (1998). CEN/TC 251 "*CEN/TC 251/N98-082, Version 1.2 Health Informatics Short Strategic Study: Enabling Technolgies - UML (Final Report)*", accessed via www.centc251.org, dated 25 May 1998.

CEN_a (1999). CEN/TC 251 Health Informatics, "*ENV13606-4:1999 Health informatics – Electronic Healthcare record communication - Part 4: messages for the exchange of information*" accessed via www.centc251.org, March 2005.

CEN_b (1999). CEN/TC 251 Health Informatics, "*ENV13606-1:1999 Health informatics - Electronic healthcare record communication - Part 1: Extended architecture*" accessed via www.centc251.org, March 2005.

CEN_c (2007). CEN/TC 251 Health Informatics, "*prEN13606-5 rev Health informatics – Electronic health record communication – Part 5: Exchange models*", part of the prEN13606:2004(E) suite accessed via www.centc251.org, 8 March 2007. This document is a pre-standard and is currently in a state flux, but is at an advanced stage of development.

CEN_d (2007). CEN/TC 251 Health Informatics, "*prEN13606-1 rev Health informatics - Electronic health record communication — Part 1: Reference model*", part of the prEN13606:2004(E) suite accessed via www.centc251.org, 8 March 2007. This document is a pre-standard and is currently in a state flux, but is at an advanced stage of development.

CEN_e (2007). CEN/TC 251 Health Informatics, "*prEN13606-2 rev Health informatics – Electronic health record communication — Part 2: Archetypes Version 2.2*", part of the prEN13606:2004(E) suite accessed via www.centc251.org, 8 March 2007. This document is a pre-standard and is currently in a state flux, but is at an advanced stage of development.

Chatfield, Christopher, (1983). "*Statistics for Technology*", third edition, published by Chapman and Hall, ISBN:1412253402 in 1983.

Ciccarese P, Caffi E, Boiocchi L, Quaglini S, Stefanelli M., (2004). "*A guideline management system*". Medinfo. 2004;2004:28-32.

Claire, G., Lhuillier N., G. Rimassa, (2005). "*A communication protocol for agents on handheld devices*" accessed via http://exp.telecomitalialab.com via the web on the August 2005.

CLSI (2005). "*How to Define and Determine Reference Intervals in the Clinical Laboratory; Approved Guideline*" published by the Clinical and Laboratory standards institute, second edition 2005. Reference C28-A2, ISBN: 1562384066.

CLSI (2006). Clinical and laboratory standards institute, "*Harmonized Terminology Database*", accessed via the web http://www.clsi.org/source/custom/termsall.cfm , May 2006.

Cockburn, A., (1999). "*An Investigation of Groupware Support for Collaborative Awareness Through Distortion-Oriented Views*" published in 1999, accessed via www.cosc.canterbury.ac.nz

Collier Christine, (2001). "*Biological Variation & Result Interpretation*" online presentation via http:/www.path.queenssu.ca, produced in 2001.

Collier, Rem William (2003). "*Agent Factory: A Framework for the Engineering of Agent-Oriented Applications*" National University of Ireland, Department of Computer Science, Faculty of Science, University College Dublin, Doctor of Philosophy Thesis, December 2002. Accessed via the web 12 October 2003

Corkill Daniel, (1989). "*Design alternatives for parallel and distributed blackboard systems*" accessed via International Conference on Multi-Agent Systems (ICMAS), accessed March 2005. This paper appeared as a chapter in Blackboard Architectures and Applications, by V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum, pages 99-136, published by Academic Press, 1989.

Corkill Daniel, (2003). "*Collaborating Software Blackboard and Multi-Agent Systems & the Future*" accessed via www.cs.umass.edu, Proceedings of the International Lisp Conference published in October 2003.

Czaja Albert J., Freese Deborah K., (2002). "*Guideline-Diagnosis and treatment of autoimmune hepatitis*", The Journal Hepatology, ISSN 1527-3350, Volume 36, Issue 2, Date: August 2002, Pages: 479-497.

Dans Peter E., (1994). "*Credibility, Cookbook Medicine, and Common Sense*", Annals of Internal Medicine, published by American College of Physicians, 1 June 1994, Volume 120, Issue 11, Pages 966-968

Dastani Mehdi, (2006). "*3APL Platform, User Guide*" accessed via http://www.cs.uu.nl/3apl, 16th November 2006.

Devlin, Thomas, (2006). "*Textbook of Biochemistry with Clinical Correlations*", sixth edition published by Wiley-Liss, ISBN 0471678082.

Dickinson Gary, Fischetti Linda, Heard Sam, (2004). "*HL7 EHR System Functional Model*", White paper, accessed via www.sanita.forumpa.it/documenti/0/100/140/148/EHR-SWhitePaper.pdf, July 2004.

Drugdigest (2005). www.drugdigest.org, the information described about the drug was detailed in the Drug Digest listing, then checked and verified using the www.drugs.com drugs online database. Accessed October 2005. These sites are up to date resources for drug information used by doctors and nurses in the healthcare sector.

Egana Aranguren, (2005). "*Ontology Design Patters for the Formalisation of Biological Ontologies*", Degree of Masters of Philosophy, University of Manchester, 2005 accessed via http://gong.man.ac.uk/docs/MPhilThesis.pdf

Ellrodt Gray, Conner Laura, Ridinger Mary, (1995). "*Measuring and improving physician compliance with clinical practice guidelines: A controlled interventional trial*". Annals of Internal Medicine, published by American College of Physicians, 1995;122(4):277–82.

Engestrom Yrjo, Miettinen Reijo, Punamaki Raija-Leena, (1999). "*Perspectives on Activity Theory*", Published by Cambridge University Press in January 1999.

FDA (2002). Food and Drug Administration "*General Principles of Software Validation; Final Guidance for Industry and FDA Staff*" version 2002, accessed via the web, http://www.fda.gov/cdrh/comp/guidance/938.html

Ferrari, Luca (2004). "*The Aglets 2.0.2 User's Manual*" published 6 October 2004 accessed via the http://sourceforge.net/projects/aglets/.

Field MJ, Lohr KN, (1992). "*Guidelines for clinical practice. From development to use*". Washington, DC: National Academy Press, 1992.

FIPA, (2002). "*FIPA Agent Message Transport Envelope Representation in Bit-Efficient Encoding Specification*", published by Foundation for Intelligent Physical Agents (FIPA) document number SC00088D, date 03 December 2002, accessed via http://www.fipa.org/specs/fipa00088/SC00088D.pdf

FIPA_a (2003). Foundation for Intelligent Physical Agents (FIPA) "*FIPA Agent Management Specification*", Document number SC00023K dated 2004/18/03, accessed via http://www. fipa.org

FIPA_b (2003). Foundation for Intelligent Physical Agents (FIPA), "*FIPA Agent Message Transport Service Specification*", Document Number SC00067F, Dated 2002/12/03, accessed via http://www. fipa.org

FIPA_c (2003). Foundation for Intelligent Physical Agents (FIPA), "*FIPA Device Ontology Specification*", Document Number SC00091E, Dated 2002/12/03, accessed via http://www. fipa.org

FIPA_d, (2003) Foundation for Intelligent Physical Agents (FIPA), Radovan Cervenka, "*Modeling Notation Source-Prometheus*", Version: 03-04-02, Dated 2003, accessed via http://www.auml.org/auml/documents/Prometheus030402.pdf

Fox J, Das S., (2000) "*Safe and Sound: Artificial Intelligence in Hazardous Applications*". AAAI and MIT Press, 2000, ISBN 978-0-262-06211-4.

Fraser_a Callum G., (2000). "*Biological variation data for setting quality specifications in laboratory medicine*" accessed via the web on http://www.westgard.com/guest12.htm, dated 2000.

Fraser_b Callum G., (2001). "*Biological variation from principles to practice*", published by the American Association for Clinical Chemistry, 2001, ISBN 10:1890883492.

Fridsma Douglas B., Gennari John H., Musen Mark A., (1996). "*Making Generic Guidelines Site Specific*", published as part of the Proceedings of American Medical Informatics Association, Fall Symposium, 1996 ;597-601.

Gangemi Aldo, Pisanelli Domenico M., Steve Geri, (1999). "*An Overview of the ONIONS Project: Applying Ontologies to the Integration of Medical Terminologies*", Data and Knowledge Engineering, vol. 31, 1999, Pages 183-220, published by Elsevier accessed via the web http://www.sciencedirect.com/science/journal/0169023X

Garson James, (2003) "*Stanford Encyclopedia of Philosophy: Modal Logic*" http://plato.stanford.edu/entries/logic-modal/, dated 2003.

GEA, (1998). "*Gigabit accelerating the standard for speed Ethernet*", Whitepaper produced by the Gigabit Ethernet Alliance, accessed www.ethernetalliance.org/technology/white_papers/gea_speed.pdf

Gene, (2000). "*Gene Ontology: tool for the unification of biology*", published by the Gene Ontology Consortium, accessed via http://www.geneontology.org/GO_nature_genetics_2000.pdf

Georgeff, M. P. and Ingrand, F. F. (1989). Decision-Making in an Embedded Reasoning System, Proc. of the Inter. Joint Conference on Artificial Intelligence, IJCAI'89, 202-206, Detroit, Mich.,USA.

GPAC (2004) "*Liver Chemistry Abnormalities in Adults–Evaluation and Interpretation*" published by Guidelines and Protocols Advisory committee in British Columbia Effective July 2004, accessed via http://bcguidelines.ca/

Greenwood Dominic, (2004). "*FIPA-The Foundation for Intelligent and Physical Agents*", 10 May 2004, of Whitestein Technologies AG, accessed via http://jade.tilab.com/papers/JADETutorialIEEE/JADETutorial_FIPA.pdf

Grimson Jane, (1997). "*Synapses Federated Healthcare Record Server*" HC1046 PowerPoint presentation, presented at the annual project review 1997.

Grimson William, Berry Damon, Grimson Jane, Stephens Gaye, Felton Eoghan, Given Peter, O'Moore Rory, (1997). "*Federated Healthcare Record Server - the Synapses paradigm*" accessed via the web on http://www.cs.tcd.ie/synapses/public/publications/mie97.doc, dated 1997

Gruber Thomas R., (1993). "*A Translation Approach to Portable Ontology Specifications*", Appeared in Knowledge Acquisition, 5(2):199-220, 1993, accessed via the web http://tomgruber.org/writing/ontolingua-kaj-1993.pdf Knowledge Systems Laboratory, Technical Report KSL 92-71

Harmening Denise M, (2003). "Laboratory *Management: Principles and Processes*", section 13, published by Prentice Hall, ISBN 013019459

Harrison Michael, (2004). "*Liver Function Tests –like the curate's egg: good in parts*" Published by Sullivan-Nicolaides-Pathology, Synopsis Issue 33, July 04 accessed via the web http://www.snp.com.au/publications/pdf/doct/synop/SYNOPSIS%2033%20article1.pdf

Helin, H. Laukkanen, M., (2002). "*Performance analysis of software agent communication in slow wireless networks*", Computer Communications and Networks, 2002. Proceedings. Eleventh International Conference, 14-16 October 2002, pages 354-361 ISBN: 0-7803-7553-X.

Hermans Björn, (1996). "*Intelligent Software Agents on the Internet: an inventory of currently offered functionality in the information society & a prediction of (near-) future developments*", Tilburg University, Tilburg, The Netherlands, produced 9th of July 1996, accessed via http://www.hermans.org/agents.

Hoffband, A.V., Pettit, J.E., Moss, P.A.H., (2001). "*Essential Haematology*", fourth edition published by Blackwell Science, ISBN 0632051531.

Holland J.H., (1996) "*Hidden Order: How Adaptation Builds Complexity*": published Addison Wesley, Reading, MA, USA in September 1996, ISBN:9780201442304.

Hubner Jomi Fred, Sichman Jaime Simao, (2003). *"Saci Programming Guide(version 0.9)"*, Laborat´orio de T´ecnicas Inteligentes Universidade de S˜ao Paulo, Escola Polit´ecnica, Laborat´orio de T´ecnicas Inteligentes, July 2003.

Ingram David, Beale T, Heard S, Kalra D, Lloyd D, Schloeffel P, (2006). "*Introducing openEHR*" accessed via http://svn.openehr.org/specification/TRUNK/publishing/openEHR/introducing_openEHR.pdf, dated Jan 2006.

Isenberg, Henry D., (1998). "*Essential Procedures for Clinical Microbiology*", Published by American Society for Microbiology (ASM) Press Washington, ISBN 1555811256.

ISO 3534-1, (1993). "*Statistics, Vocabulary and Symbols*", Part 1, Probability and General Statistical Terms, ISO, 1993, p. 34.

Jaeger Jens Joergen, Hedegaard Hanne, (2004). "*About blood tests*", accessed via http://home3.inet.tele.dk/omni/alttest.htm, March 2004.

Johnson P, Tu S, Jones N. (2001). "*Achieving reuse of computable guideline systems*". Medinfo. 2001;10(Pt 1):99-103.

Johnston David E., (1999). "*Special Considerations in Interpreting Liver Function Tests*" American Family Physician, published by the American Academy of Family Physicians Vol. 59 Issue No 8, April 15, 1999, accessed via http://www.aafp.org/afp/990415ap/2223.html

Jung Benjamin, Grimson Jane, (1999). "*Synapses/SynEx goes XML*", accessed via the Trinity Web Site https://www.cs.tcd.ie/synapses/public/publications/mie_aug99_v.pdf

Kakas Antonis, Mancarella Paolo, Sadri Fariba Stathis Kostas Toni Francesca, (2003). "*The KGP Model of Agency*" accessed via the web 12 March 2003.

Kaptelinin Victor, (1997). "*Activity Theory: Basic Concepts and Applications*", accessed via http://acm.org/sigchi/chi97/proceedings/tutorial/bn.htm, published in 1997.

Kavanagh Matt, Price Susan, (2002). "*The quest for a computerized guideline standard: The process, its history, and an evaluation of the most common and promising methods used today"*, Capstone Project 2002.

Khamis Abdelaziz, Aboul-Ela Magdy, Atwany Mohamed, (2004). "*A Framework for Multi-agent Collaboration*", The scientific 11th conference for information and computer technology, Egyptian Society for Information Systems and Computer Technology (ESISACT), Cairo, 10-12 Feb. 2004.

Koes M., Nourbakhsh I., Sycara K., (2004). "*Communication Efficiency in Multi-Agent Systems*", proceedings of IEEE International Conference on Robotics and Automation (ICRA) 2004, Vol. 3, May, 2004, pp. 2129 - 2134.

Kumar Parveen, Clarke Michael, (2002). "*Clinical Medicine*", 5th edition, published by Saunders (W.B.) Co Ltd, 31 Jul 2002. ISBN 0702025798.

Lacher DA., (1990). "*Relationship between delta checks for selected chemistry tests*" Clinical Chemistry, Vol 36, 2134-2136, printed in 1990 by American Association for Clinical Chemistry, http://www.clinchem.org/cgi/reprint/36/12/2134.pdf

Lander, Susan E., Staley, Scott M., Corkill, Daniel D., (1996). "*Designing Integrated Engineering Environments: Blackboard-Based Integration of Design and Analysis Tools*", Concurrent Engineering, Vol. 4, No. 1, 59-71 (1996).

Lenat, D. B. (1978). "*The ubiquity of discovery*" published by Artificial Intelligence, Volume 9, pages 257-285.

Leont'ev A.N, (1981). "*Problems of the Development of the Mind*" Published in Moscow in 1981 by Progress Publishers, accessed via http://www.comnet.ca/~pballan/Leontyev1981chapt1.htm.

Levinson, Warren, Jawetz, Ernest, (2002). "*Medical Microbiology & Immunology*", seventh International edition published by McGraw-Hill, ISBN 0071212361.

Little Paul, Everitt Hazel, Williamson Ian, Warner Greg, Moore Michael, Gould Clare, Ferrier Kate, Payne Sheila, (2001). "*Preferences of patients for patient centred approach to consultation in primary care: observational study*", BMJ 2001;322;468- doi:10.1136/bmj.322.7284.468 http://bmj.com/cgi/content/full/322/7284/468 accessed 2006.

Luck Michael, McBurney Peter, Preist Chris, (2004). "*Agent Technology: Enabling Next Generation Computing: A Roadmap for Agent-Based Computing*" Accessed via AgentLink web site http://www.agentlink.org/roadmap/ 2004 ISBN 0854 327886.

Luger George, F., (2002) "*Artificial Intelligence*" published by Addison Wesley Fifth Edition, ISBN 0-321-26318-9. Chapter 8, pgs 277-332, 21 Oct 2004.

Luger_a, (2002) Luger George, F., "Artificial Intelligence" published by Addison Wesley Fifth Edition, ISBN 0-321-26318-9, 21 Oct 2004.

Luger_b, (2002) Luger George, F., "*Artificial Intelligence*" published by Addison Wesley Fifth Edition, ISBN 0-321-26318-9. Chapter 17 pgs 833-834, 21 Oct 2004.

Luger_c, (2002) Luger George, F., "*Artificial Intelligence*" published by Addison Wesley Fifth Edition, ISBN 0-321-26318-9. Chapter 1 pgs 3-32, 21 Oct 2004.

Marshall William J., (2000) "Clinical Chemistry", Fourth edition published in 2000 by Mosby, ISBN 0723431590.

Marshall William.J, Bangert Stephen K, (1995). "*Clinical Biochemistry, Metabolic and Clinical Aspects*" ISBN 0443043418, published in 1995 by Churchill Livingstone.

Mason Pamela, (2002). "*Drug and food interaction: Food and Medicine*", published by The Pharmaceutical Journal, October 2002, Volume 269, http://www.pjonline.com/pdf/cpd/pj_20021019_drugfood.pdf.

McLoughlin Vivienne, Millar John, Mattke Soeren, Franca Margarida, Jonsson Pia Maria, Somekh David and Bates David, (2006) "Selecting indicators for patient safety at the health system level in OECD countries" International Journal for Quality in Health Care 2006 18(Supplement 1):14-20

Medicinenet (2006). Image of the human liver obtained via http://www.medicinenet.com/images/Liver.jpg accessed August 2006.

Medicinenet_a, (2006). Medicine dictionary, information source accessed via http://www.medicinenet.com accessed August 2006.

Myers Gary L., Kimberly Mary M., Waymack Parvin P., Smith Jay, Cooper Gerald R., Sampson Eric J., (2000). "*A Reference Method Laboratory Network for Cholesterol: A Model for Standardization and Improvement of Clinical Laboratory Measurements*" Published by Clinical Chemistry 46:11 1762-1772 (2000).

Myers, K.L., (1997), "*User Guide for the Procedural Reasoning System*", Artificial Intelligence Center, Technical Report, SRI International, Menlo Park, CA.

NAAC (2005). National Anemia Action Council, "*Anemia and Critical Illness*", accessed via http://www.anemia.org/patients/educationsheets/critical_illness.pdf, 2005.

NGC (2004). The National Guideline Clearinghouse "*National Guideline Clearinghouse*™", accessed via the web http://www.guideline.gov/, accessed May 2004.

NHLBI (2004) The National Heart, Lung, and Blood Institute (NHLBI) "*Recommendations regarding public screening for measuring blood cholesterol*" accessed via http://www.nhlbi.nih.gov/guidelines/cholesterol/chol_scr.pdf, accessed March 2004.

NHLBI_b (2004). "*Encoding the National Cholesterol Education Program*" accessed via the web at http://www.nhlbi.nih.gov/guidelines/cholesterol/atglance.htm, accessed March 2004.

NICE (2006), "*Guideline-Anaemia management in people with chronic kidney disease-Quick reference guide*" published by National Institute for Health and Clinical Excellence, Issue date: September 2006, ISBN 1-84629-285-9, accessed via http://www.nice.org.uk.

NKF (2002), "*K/DOQI Clinical Practice Guidelines for Chronic Kidney Disease: Evaluation, Classification, and Stratification*", The National Kidney Foundation

Kidney Disease Outcomes Quality Initiative, accesses via http://www.kidney.org/professionals/KDOQI/guidelines_ckd/toc.htm

NLM (2006). National Library of Medicine USA "*Unified Medical Language System*" accessed via http://www.nlm.nih.gov/pubs/factsheets/umls.html, accessed October 2006

Norling Emma (2004). "*Folk Psychology for Human Modeling: Extending the BDI Paradigm*". In proceeds of Autonomous Agents and Multi-agent Systems AAMAS conference, 2004.

Noy Natalya F., McGuinness Deborah L., (2001). "*Ontology Development 101: A Guide to Creating Your First Ontology*" http://protege.stanford.edu/publications/ontology_development/ontology101.pdf. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05. March 2001.

OMG (2002). Object Management Group (OMG), "*Common Object Request Broker Architecture: Core Specification*" accessed via http://www.omg.org, November 2002 Version 3.0.

Oosterhuis Wytze P., Bruns David E., Watine Joseph, Sandberg Sverre, Horvath Andrea R., (2004). "*Evidence-Based Guidelines in Laboratory Medicine: Principles and Methods*", published in Clinical Chemistry, year 2004, Volume50 pages 806-818.

Oosterhuis Wytze P., Ulenkate Herman J.L.M., and Goldschmidt Henk M.J., (2000). "*Evaluation of LabRespond, a New Automated Validation System for Clinical Laboratory Test Results*", Clinical Chemistry 46:11, 1811–1817 (2000)

Overhage JM, Tierney WM, Zhou XH, McDonald CJ., (1997). "*A randomized trial of Corollary Orders to Prevent Errors of Omission.*" J Am Med Inform Assoc 1997;4(5):364–375.

Oxford (2005). Oxford Dictionary 7th edition, Clarendon Press, p. 253, 2005

Padgham Lin, Shehory Onn, Sterling Leon, Strum Arnon, (2005). "*Methodologies for Agent Orientated Software Engineering EASSS2005*", 7th Agent Summer School proceedings Utrecht, The Netherlands, 18-22 July, 2005

Parks David, (1997). "*Agent-Oriented Programming: A Practical Evaluation*", 515 Soda Hall University of California, Berkeley 94720, accessed via http://www.cs.berkeley.edu/~davidp/cs263/index.html, dated 12May 1997.

Peleg_a (2001) Peleg Mor, Ogunyemi O, Tu S, Boxwala A, Zeng Q, Greenes RA, Shortliffe EH.. "*Using features of Arden Syntax with object-oriented medical data models for guideline modeling*", published by Journal of the American Medical Informatics Association, 2001 (Supplement 8).

Peleg_b (2001) Peleg Mor, Boxwala Aziz A., Bernstam Elmer, Tu Samson, Greenes Robert A., Shortliffe Edward H., (2001). "*Sharable Representation of Clinical Guidelines in GLIF: Relationship to the Arden Syntax*", dated October 2001.

Peleg_c (2004) Peleg Mor, Boxwala Aziz, Tu Samson, Wang Dongwen, Ogunyemi Omolola, Zeng Qing, (2004). "*GLIF: Guideline Interchange Format 3.5 Technical Specification*", May 4, 2004

Peleg_d (2003) Peleg Mor, Tu Samson, Bury Jonathan, Ciccarese Paolo, Fox John, Greenes Robert A., Hall Richard, Johnson Peter D., Jones Neill, Kumar Anand, Miksch Siliva, Quaglini, Seyfang Andreas, Shortliffe Edward H., Stefanelli Mario, (2003). "*Comparing Computer-interpretable Guideline Models: A Case-study Approach*", accessed via Journal of the American Medical Informatics Association Volume 10 Number 1 Jan / Feb 2003.

Petersen Hyltoft, (2005). "*Making the Most of a Patient's Laboratory Data:Optimisation of Signal-to-Noise Ratio*", Clin Biochem Rev Vol 26 November 2005 pages 91-96.

Pierangelo Bonini, Mario Plebani, Ferruccio Ceriotti, Francesca Rubboli, (2002). "*Errors in Laboratory Medicine*" published by Clin. Chem., May 2002; 48: 691 - 698

Plebani Mario, Carraro Paolo, (1997). "*Mistakes in a stat laboratory: types and frequency*", Published by Clinical Chemistry 1997;43:1348-1351 accessed via web site on 3/10/11.

Pokahr_a, Alexander, Lars Braubach, Lamersdorf Winfried (2005). "*A Flexible BDI Architecture Supporting Extensibility*", Published in the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-2005). http://vsis-www.informatik.uni-hamburg.de/projects/jadex/papers/architecture.pdf

Pokahr_b, Pokahr, Alexander, Lars Braubach, (2005). "*Jadex User Guide*", Document notes issued with Release 0.932 dated the 10th May 2005 accessed via http://visis-www.informatik.uni-hamburg.de/projects.jadex.

Pokahr_c, Alexander, Lars Braubach, W. Lamersdorf, (2003). "*Jadex: Implementing a BDI-Infrastructure for JADE Agents*" accessible via http://vsisls1.informatik.uni-hamburg.de/getDoc.php/publications/124/pokahrbraubach2003jadex-exp.pdf, Published in http://exp.telecomitalialab.com exp – Vol. 3 - No. 3 - September 2003

Pressman_a (1997) Pressman R.S.. "Software Engineering: A Practitioner's approach" published by McGrawHill, fourth edition 1997, ISBN 0-07-709411-5 pgs 4-5, in 1997.

Pressman_b (1997) Pressman R.S., "*Software Engineering: A Practitioner's approach*" published by McGrawHill, fourth edition, ISBN 0-07-709411-5 Chapter 13, pgs 357-386, in 1997.

Pressman_c (1997) Pressman R.S., "*Software Engineering: A Practitioner's approach*" published by McGrawHill, fourth edition, ISBN 0-07-709411-5 Chapter 19, pgs 567-593, in 1997.

Purves I, (1995). "*Computerised Guidelines in Primary Health Care: Reflections and Implications*", Health Telematics for Clinical Guidelines and Protocols editied by Gordon C, 1995 pages 57-74.

Raja Anita, Lesser Victor, (2004). "*Meta-level Reasoning in Deliberative Agents*". In Proceedings of the International Conference on Intelligent Agent Technology (IAT 2004). Beijing, China, September 2004. pp 141-147.

Rao A., Georgeff M., (1995). "*BDI Agents: From Theory to Practice*" proceedings of the first international conference on multi agent systems (ICMMAS-95) San Francisco USA 1995.

Rao Gopal, Crook M., Tillyer M.L., (2003) "*Pathology tests: is the time for demand management ripe at last?*", Journal of Clinical Pathology 2003;56:243-248 BMJ Publishing Group & Association of Clinical Pathologists.

Rector Alan, Rogers Jeremy, (2005). "*Ontological & Practical Issues in using a Description Logic to Represent Medical Concepts: Experience from GALEN*", School of Computer Science, The University of Manchester ,Preprint Series, CSPP-35 accessed via http://www.cs.manchester.ac.uk/cspreprints/PrePrints/cspp35.pdf

Rector Alan, Rogers Jeremy, Bittner Thomas, (2006). "*Granularity, scale and collectivity: When size does and does not matter*", Journal of Biomedical Informatics 39 (2006) 333–349, accessed via http://www.sciencedirect.com

Ricci Alessandro, Omicini Andrea, Denti Enrico, (2002). "*Activity Theory as a Framework for MAS Coordination*", DEIS, Universita' di Bologna, Cesena, Italy, presented at the ESAW 2002, 17 September 2002 in Madrid.

Rodney Jackson, Gene Feder, (1998). "Guidelines for clinical guidelines should distinguish between national and local production" published by BMJ 1998;317:427-428

Sadava David, Heller Craig, Orians Gordon H., Purves William K., Hillis David, (2006). "*Life: The Science of Biology*" 8th Edition, published by Sinauer Associates Eighth Edition. http://www.sinauer.com. ISBN 0716798565, December 2006.

Schloeffel Peter, (2003) "*Clinical communication using HL7 and the EHR*", *open*EHR Foundation Ocean Informatics, EHRcom, HL7 Australia Conference, 5 Mar 2003, accessed via http://www.HL7.org, March 7, 2007.

Schumacher Michael, (2001). "*Object Coordination in Multi Agent System Engineering: Design and Implementation*", published by Springer, ISBN 3450419829, Jan 2001

SEER, (2005). National Cancer Institute, "*Surveillance, Epidemiology and End Results (SEER) Program*", Emory University, Atlanta SEER Cancer Registry, Atlanta, Georgia, U.S.A, accessed January 2005.

Serdar Muhittin A., Kurt Ismail, Ozcelik Fatih, Urhan Muammer, Ilgan Seyfettin, Yenicesu Mujdat, Kenar Levent and Kutluay Turker, (2001). "*A Practical Approach to Glomerular Filtration Rate Measurements: Creatinine Clearance Estimation Using Cimetidine*" Annals of Clinical & Laboratory Science 31:265-273 (2001), accessed via http://www.annclinlabsci.org/cgi/reprint/31/3/265

Shahar, Y., Miksch, S., and Johnson, P., (1998). "*The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines*". Artificial Intelligence in Medicine (14): 29-51, 1998.

Shoham Yoav, (1993). "*Agent-oriented programming*". Artificial Intelligence, 60:51–92, 1993.

Stern M., Brennan S., (1994). "*Medical audit in the hospital and community health services*" London: Department of Health 1994.

Stewart Moira, (2006). "*Towards a global definition of patient-centred care*", Editorial comments, BMJ 2001;322;444-445, http://bmj.com/cgi/content/full/322/7284/444 accessed 2006.

Stuckenschmidt H., van Harmelen F., Serafini L., Bouquet P., Giunchiglia F., (2004). "*Using C-OWL for the Alignment and Merging of Medical Ontologies*", Proceedings of the First International Workshop on Formal Biomedical Knowledge Representation ({KRMed'04})", Whistler, BC, Canada, 1 June 2004, accessed via http://sunsite.informatik.rwth-aachen.de/Publications/CEUR WS/Vol-102/stuckenschmidt.pdf

Sunderman William F, (1992). "*The History of Proficiency Testing and Quality Control*", Clinical Chemistry, Vol.38, No. 7, 1992 1205-1209

Sutton DR, Fox J., (2003). "*The Syntax and Semantics of the PROforma guideline modeling language*". J Am Med Inform Assoc. 2003 Sep-Oct;10(5):433-43.

Tenenbaum, Joshua. B., Griffiths, T. L., Kemp, C., (2006). "*Theory-based Bayesian models of inductive learning and reasoning*". Trends in Cognitive Sciences, 2006, 10(7), 309-318 Massachusetts Institute of Technology.

TMAP (2006). The Medical Algorithms Project, "*Chapter 40. Unit Conversions*", Accessed via http://www.medal.org/visitor, accessed August 2006.

Tu SW, Musen MA., (2001). "*Modeling Data and Knowledge in the EON Guideline Architecture*". Proc. MedInfo 2001, London, UK, 280-284. 2001.

Turban_a (2005) Turban E., Aronson J.E., Liang TP., "*Decision Support Systems and Intelligent Systems*" published in 2005 by Pearson Prentice Hall, Seventh Edition, ISBN 0-13-123013-1, Chapter 11 pgs 616-617.

Turban_b (2005) Turban E., Aronson J.E., Liang TP., "*Decision Support Systems and Intelligent Systems*" published in 2005 by Pearson Prentice Hall, Seventh Edition, ISBN 0-13-123013-1, Chapter 13, pgs 700-742.

Turban_c (2005) Turban E., Aronson J.E., Liang TP., "*Decision Support Systems and Intelligent Systems*" published in 2005 by Pearson Prentice Hall, Seventh Edition, ISBN 0-13-123013-1, Chapter 10, pgs 538-570.

Turban_d (2005) Turban E, Aronson J.E., Liang T-P. "*Decision Support Systems and Intelligent Systems"*. Published in 2005 by Pearson Prentice Hall, Seventh edition, ISBN 0-13-123013-1. Chapter 2, pgs 36-98.

Valdiguie PM, Rogari E., Philippe H., (1992). "*VALAB: Expert System for Validation of Biochemical Data*", Clinical Chemistry 38/1, 83-87 (1992).

Van Bemmel J., Musen M.A., (2002). "*Handbook of Medical Informatics*", section 13 and 16, published by Springer, ISBN 3540633510.

Vaucher Jean, Ncho Ambroise, (2003). "*JADE Tutorial and Primer*", September 2003, accessed via the web http://www.iro.umontreal.ca/~vaucher/Agents/Jade/JadePrimer.html

Vinoski Steve, (1997). "*CORBA: Integrating Diverse Applications within Distributed Heterogeneous Environments*", IEEE Communications Magazine, Vol. 35, No. 2. Steve Vinsoki is an employee of IONA Technologies and the document was published by the IEEE 1997.

W3C (2002). W3C Authors, "Extensible Markup Language (XML) 1.0", Second edition accessed via http://www.w2.org/TR/REC-xml, March 2002.

Westgard_a (1997). Westgard JO, Barry Patricia L., "*Cost-Effective Quality Control: Managing the Quality and Productivity of Analytical Processes*". ISBN10: 0915274353, AACC Press published in July 1997.

Westgard_b (2003). Westgard JO, "*Six Sigma Quality Management and Desirable Laboratory Precision*" accessed via the web http://www.westgard.com/essay35.htm, published in 2003.

Westgard_c (2000). Westgard JO "*QC-The Levey-Jennings Control Chart*" accessed via the web http://www.westgard.com/lesson12.htm, published in 2000.

Westgard_d (2002). Westgard JO, "*Basic Qc Practices: Training in Statistical Quality Control for Healthcare Laboratories?*", published by AACC Press, 2nd edition in January 2002, ISBN101886958173.

Westgard_e (2005). Westgard JO, "*Multirule and "Westgard rules": What are they?*" sourced via web site at http://www.westgard.com/mltirule.htm, published 2005

Witte David L., VanNess Sue Ann, Angstadt Debbie S., and Pennell Beverly J., (1997). "*Errors, mistakes, blunders, outliers, or unacceptable results: how many?*" Published by Clinical Chemistry 1997;43:1352-1356 accessed via web site on 3/10/11.

Wooldridge M. (2000). "*Reasoning about Rational Agents, Intelligent Robots and Autonomous Agents*". The MIT Press, Cambridge, Massachusetts, 2000, ISBN10: 0262232138.

Worman Howard J. (1999). "*The Liver Disorders Sourcebook*", published by McGraw-Hill; 1 edition (August 1, 1999), ISBN-10: 0737300906.

Young DS., (2000). "*Effects of drugs on clinical laboratory tests*" volume 2, 5th ed. Washington DC: American Association for Clinical Chemistry AACC Press, 2000. Web access http://www.aacc.org/AACC/.

Appendix

A

Abbreviations and glossary

A.1 Abbreviations.

ACL	Agent Communication Language
ADF	Agent definition files
ADL	Archetype Description Language
AID	Agent-Identifier, a list of arguments used to uniquely identify an agent
ALT	Alkaline aminotransferase
ANSI	American National Standards Institute
AOP	Agent Orientated Programming
ASK-agents	Autonomous Socialising Knowledge agents
ASTM	American Society for Testing and Materials
AUML	Agent Unified Modeling Language
BMI	Body Mass Index
BOA	Basic Object Adapter
BRF	Belief Revision Function
cADL	Constraint Archetype Definition Language
CBR	Case-based reasoning
CEN	European Committee for Standardization (Comité Européen de Normalisation)
CHD	Coronary heart disease
CIG	Computer Interpretable Guideline
CLSI	Clinical and Laboratory Standards Institute, formally known as NCCLS
CORBA	Common Object Request Broker Architecture
CV	Coefficient of Variation
CV_A	Analytical coefficient of variation
CV_G	Between-subject biological variation
CV_I	Within-subject biological variation
dADL	Data Archetype Definition Language
DBMS	Database Management System
DF	Agent Directory Facilitator
EHR	Electronic Healthcare Record
EN	Européen de Normalisation, a CEN standard

FIPA	Foundation for Intelligent Physical Agents
GGT	Gamma-Glutamyl Transpeptidase
GLIF	Guideline Interchange Format
HDL	High Density Lipoprotein
HL7	Health Level Seven
HSE	Health Service Executive
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
IDL	Interface Definition Language
IFCC	International Federation of Clinical Chemistry
II	Index of Individuality
IIOP	Internet Inter-ORB Protocol
IS	Information Systems
IT	Information Technology
IUPAC	International Union of Pure and Applied Chemistry
JADE	Java Agent Development Environment
JDE	Java Development Environment
JDK	Java Development Kit
KQML	Knowledge Query and Manipulation Language
LDL	Low Density Lipoprotein
LFT	Liver Function Test
LIS	Laboratory Information System
MaSE	Multi-agent Software Engineering
MDE	Multi-Agent System Development Environment
MLM	Medical Logic Module
MTP	Message Transport Protocol
MTS	Message Transport Service
ODP	Open Distributed Processing
OQL	Object Query Language
ORB	Object Request Broker
QAP	Quality Action Plans

RIM	Reference Information Model
RMI	Remote Method Invocation
RMS	Root Mean Square
SACI	Simple Agent Communication Infrastructure
SD	Standard Deviation
SDI	Standard Deviation Index
SOP	Standard Operating Procedure
SQL	Structured Query Language
TNM	Task-Network Models
UML	Unified Modelling Language
XML	Extensible Mark-up Language

A.2 Glossary.

Analysis – the procedural steps performed that enable the measurement of the amount of analyte in a specimen.

Analyte – the constituent of the specimen to be measured.

Applet – a software component that runs in the context of another program and usually performs a very narrow function that has no independent use.

Archetype – is the formal definition of prescribed combinations of the building-block classes defined in the Reference Model for particular clinical domains or organisations. An archetype is a formal expression of a distinct, domain-level concept, expressed in the form of constraints on data whose instances conform to some class model, known as a reference model [CEN_e, 2007].

Batch – a group of similar sample types which are all going to undergo the same procedure.

Batch Run – the consecutive processing of a group of similar sample types which are all going to undergo the same procedure.

Between-subject biological variation – the difference between the homeostatic setting points of individuals.

Bias (of measurement) – the difference between the results of measurement and the true value of the measured quantity, in practice, bias is the difference between the results and an estimate of the true value.

Clinician – a term used to describe doctors, physicians, nurses, therapists or other healthcare professionals who are licensed to deal with patients.

Coefficient of variation (CV) – a measure of relative precision calculated as a standard deviation (SD) divided by its mean and often multiplied by 100 and expressed as a percentage.

Control program – see Inference engine.

Coronary heart disease – a disease of the heart itself caused by the accumulation of athermanous plaques within the walls of the arteries that supply the myocardium.

Creatinine – used to evaluate kidney function and can be measured through a urine sample or a serum sample.

Customised functions – used to encode information and knowledge written in guidelines into CIG's

Decision support systems – are a class of computer-based information systems or knowledge based systems that, in very different manner, support decision making activities.

Delta-check – a technique for quality assurance, using patient data, based on investigating the differences between two values on individuals that are greater than previously set value.

Diagnosis – a process that involves identifying disease by investigating the symptoms.

Domain – a field of study, and knowledge of the objects and relationships in that field of study.

Error recover rate – the identification of a result that was wrongly confirmed as plausible for a patient.

False negatives – results from diseased people that are not unusual.

False positives – results from well people that are not unusual.

Gaussian distribution – a symmetrical bell-shaped distribution with a defined mathematical function.

Golden-pages – also known as yellow-pages, refers to a directory which permits the searching using multiple parameters or categories, such as service, ownership etc.

Groupage – refers to the grouping together of several compatible entities by virtue of some shared characteristics.

Guideline encoder – the person converting the guideline into the agent.

Guideline plan – refers to a workflow path contained within the guidelines to achieve some objective.

Hereditary heart disease – heart disease caused by unavoidable genetic factors since birth.

Homeostatic set point – a range around which a specific analyte result from an individual varies over time.

Hypertensive heart disease – heart disease caused by high blood pressure.

Index of Individuality – the ratio of within-subject to between-subject variation most often calculated as the simple ratio of the biological components of variation.

Individuality – the property of being characteristic of a particular person.

Inference engine – software acting on the knowledge base and provides some decision-making or reasoning process with controls

Inference intelligence – the process of executing logic where more than one exact result is possible and some deduction is necessary.

Instantiation – The act of creating an 'instance' of a generic unit, module or software by replacing its formal parameters by a set of matching actual parameters.

Introspection – the self-observation and reporting conscious inner thoughts, desires and sensations.

Islands of information – modules of software which are not (or cannot be) integrated together, as a result of networking or interfacing issues, are considered isolated information sources.

JACK – a Belief, Desire and Intention (BDI) agent system using a component-based approach. Based on the JACK Agent Language (JAL) an extension of Java.

Knowledge engineering – work in knowledge representation focusing on representational formalism (e.g. explicit description of a human, or glucose test) or on the information to be encoded.

Liver Function Test – Liver function tests measure various chemicals in the blood made by the liver. An abnormal result indicates a problem with the liver, and may help to identify the cause. Further tests may be needed to clarify the cause of the liver problem.

Modularity – the property of computer programs that measures the extent to which they have been composed out of separate parts called modules.

Monitoring – following over time which usually involves reviewing laboratory test results over time to assess change in serial results.

Pattern matching – a method used by an inference engine to direct searches through the knowledge base.

Perception – the process of acquiring, interpreting, selecting, and organizing sensory information.

Real-world – used to describe the physical reality of everyday life which everyone experiences.

Sample – refers to a portion of a specimen which is used for analysis, many samples may be taken from a single specimen, for example a blood specimen may be divided up into clotted or unclotted (using anti-coagulant agents) samples for specific types of analysis to take place.

Sample Matrix – refers to the substance or base from which the control material is prepared in addition to all the additives such as spiking materials, preservatives, etc. added to make the product desirable to the user.

Screening – the identification of unrecognized disease or defect.

Serology – a blood test to detect the presence of antibodies against a microorganism.

Serum – refers to blood plasma in which clotting factors (such as fibrin) have been removed naturally by allowing the blood to clot prior to isolating the liquid component.

Snowmed – (Systematized Nomenclature of Medicine) is a systematically organised computer processable collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, micro-organisms, pharmaceuticals etc.

Sociality – refers to the ability to form societies and be part of an organization.

Software paradigm – a framework to support and orchestrate attributes, methods and motivations.

Standard deviation – the square root of the variance; an estimate of the dispersion of a series of measurements on the same analyte.

Stratification – dividing a sample into homogeneous sub-samples based on one or more characteristics of the population

Synapses – a project completed under the EU 4th Framework Health Telematics Programme, where a system was developed to retrieve distributed medical information from different sources present it to a user in a context rich format. The structure was initially designed as a feeder system.

Tuple – a data object that holds several objects TUPLE [X, Y, Z].

Validation – checking if a statement is true.

White-pages – refers to a directory which permits the searching using a single parameter such as name or AID.

Within subject biological variation – are inherent biological variation around the homeostatic setting point.

World-model – a constructed simplified model of the real-world environment.