

# The PROforma guideline specification language: progress and prospects

Bury, J., Fox, J., Sutton, D.

*Advanced Computation Laboratory, Imperial Cancer Research Fund, London, UK.*

*{jb,jf,ds}@acl.icnet.uk*

**Abstract.** Medical guidelines are constructed with the aim of assisting clinicians in making decisions that are informed by the best available medical evidence. In order to achieve this aim, they must be disseminated in a form that makes them easy for clinicians to use and easy for domain experts to critique. Furthermore, the language in which they are expressed must facilitate their transfer between institutions and their adaptation to local conditions. This paper describes PROforma, a knowledge representation language that attempts to meet these desiderata.

PROforma is formal knowledge representation language designed to capture the content and structure of a clinical guideline in a form that can be interpreted by a computer. The language embodies many contemporary themes in machine interpretable guideline representation schemas whilst retaining some distinctive features. This paper describes the key features of the PROforma language in the context of recent trends and developments in the field of electronic guideline representation formats. We describe our experiences in applying PROforma to a range of clinical decision making and workflow problems, and the benefits and limitations of the current language specification are discussed. Finally, we outline plans for the further refinement of PROforma.

## **Keywords :**

Decision support systems, Clinical guidelines, Knowledge engineering, PROforma.

## **1 Introduction**

This paper describes the key features of the PROforma language in the context of recent trends and developments in the field of electronic guideline representation formats. The structure of the paper is as follows: Section 2 outlines the problems that clinical guidelines address and section 3 explains the paradigms that have arisen for the expression of such guidelines. Section 4 describes the theoretical basis of PROforma, section 5 describes its implementation in software, and section 6 describes our experience of PROforma in use. Section 7 describes some of the major challenges that PROforma is designed to meet, and section 8 describes our plans for extension of the language. Finally section 9 contains a discussion of the issues raised in the paper and outlines our conclusions.

## 2 Guidelines and workflow

The explosion of interest in developing and publishing clinical guidelines, both using conventional media and on the World Wide Web, is an indicator of the growing importance attached to clinical guidelines in medicine. Kohn et al.[1] describe the recent estimate from the US Institute of Medicine that there may be as many as 98,000 unnecessary deaths per annum resulting from avoidable clinical error, which suggests there is a genuine need for such guidelines. The vast majority of clinical practice guidelines are published as text, and typically include criteria describing their applicability to particular groups of patients, the recommended processes of care, appropriate use of materials and procedures and so on, as well as providing ancillary information such as supporting evidence. Clinical guidelines are generally concerned with two main areas that directly affect the quality and effectiveness of patient care: the quality of clinical decision-making, and the correct and timely management of clinical tasks. A clinical guideline thus incorporates information relating to both decision-making and workflow management.

### 2.1 *Decision-making: What to believe and what to do*

Decision theory describes two main classes of decision: those concerning what to *believe* (e.g. what is wrong with a patient, how serious a disease is, the possible prognosis) and those concerning what to *do* (e.g. whether or not a patient should be referred for specialist care, what treatment or treatment strategy is appropriate, whether tests or investigations are required). Both of these classes of decision are vulnerable to many kinds of error. Errors in diagnosis or prognosis decisions can result from the absence of critical information about a patient, or difficulties in assessing relative strengths of evidence for example. Errors in treatment and other management actions can result from incorrect decisions about eligibility or failures to allow for possible drug interactions, etc.

### 2.2 *Task management: Who does what, when, how and where*

Even assuming that all clinical decisions are taken correctly, there are still countless ways in which errors can be made, and which can be costly in terms of the efficacy or efficiency of patient care. Tasks may be inadvertently omitted, carried out too late or adverse events may be missed. More complex clinical procedures such as care pathways or chemotherapy protocols may be carried out over an extended timescale by multiple healthcare providers, resulting in further potential hazards e.g. through a failure to keep sufficiently complete clinical records or ensure that staff in the care team are kept informed about actions that have been carried out or are planned.

### 2.3 *Delivering Clinical Guidelines*

While the systematic preparation and publication of clinical guidelines is of great importance, it is only a first step towards ensuring consistent compliance with the standards of care represented in those guidelines [2]. A guideline may set out current best-practice in great detail, but if clinicians do not have time to access the guidelines, or fully absorb their content and apply it correctly in their decision-making, or the clinical organisation fails to ensure that all the required tasks are carried out in a timely manner, then the objectives of the guideline may not be reflected in clinical outcomes.

From an AI perspective there is an obvious alternative to publishing guidelines solely in human readable form such as text, tables or flow diagrams, which is to formalise the medical knowledge contained in these guidelines in a form that a computer can apply to support clinicians in their routine work. The *PROforma* guideline specification language and its associated software attempt to provide such a formal framework. In this paper we describe the *PROforma* language in the context of recent trends in guideline representation,

review the experience we have gained in applying PROforma technology in a variety of clinical domains, and present our plans for the future development of PROforma.

### **3 Paradigms for representing clinical guidelines and workflow**

#### *3.1 The procedural approach*

Early computerised clinical support systems, including decision support systems such as statistical decision aids and clinical algorithms, whether expressed as conventional programs or in visual notations such as flow diagrams, did not separate the clinical knowledge from the computation details of how that knowledge was to be delivered. Although such systems can be satisfactory, many limitations of the procedural approach have been identified in the AI and knowledge engineering literature.

First of all users, particularly non-programmers, find it difficult to comprehend the underlying clinical logic by simple inspection of a computer program. Second, maintaining the guideline software as medical knowledge increases is notoriously difficult, and such systems lack *reusability* - the knowledge they embody cannot be readily exported to other systems in a modular form. As a consequence it is increasingly recognised that a better way of representing medical knowledge is in *declarative* form, which allows knowledge of medical concepts such as diseases, drugs and tests to be clearly separated from the reasoning and problem-solving processes that use that knowledge in particular medical contexts.

#### *3.2 Rule based systems*

Rule based expert systems such as MYCIN [3] and Oncocin [4] reflected the recognition of the importance of declarative knowledge representation and pioneered a significant change in the development of clinical decision support systems in the 1970s and '80s. Such systems clearly separated domain-specific knowledge expressed as sets of rules from the generalised inference engine, typically a backward-chaining or forward-chaining system, which would apply those rules. This represented a paradigm shift in the construction of decision support systems, and promised improved readability, modularity and reusability of the knowledge bases. However, despite the popularity and technical elegance of rule based systems, and the enhanced readability they have brought, they have been less successful in achieving modularity and reusability of knowledge. In practice rules can be written in a procedural way and frequently depend upon being carefully crafted to ensure that they are only applied in specific situations making their transfer to other applications difficult [5] and potentially hazardous, as demonstrated in studies with the Arden syntax [6], which can be viewed as a hybrid of simple rule-based and more traditional procedural methods.

#### *3.3 Task based systems*

The limitations of rule-based systems have led to the recent emergence of a further paradigm shift in the development of clinical knowledge systems. Here, rule-based and task-based formalisms are combined to represent clinical processes. How a particular logical condition should be interpreted and acted upon depends, for example, on whether that rule represents part of a diagnostic process, a precondition for a particular clinical intervention, or a contraindication to a particular drug therapy. Task-based systems therefore attempt to contextualise rules within explicit and intuitive models of the clinical process with the aim of combining the programmatic richness of a procedural representation with the logical clarity of declaratively expressed knowledge.

Use of an intuitive representation of the clinical process has at least two potential advantages. Firstly, building the knowledge base should be an easier process for domain experts to participate in, given that tasks provide a higher-level and more intuitive set of

design primitives than conventional programming languages or languages based on rules. Secondly, systems built around this approach have the potential to integrate into the clinical environment and workflow more easily, based as they are around primitives which correlate with real world concepts and processes familiar to users.

*PROforma* [7] is one example of a task-based guideline representation format. Other schemas that embody some or all of the features of the paradigm include the Guideline Interchange Format (GLIF), developed by the Intermed Collaboratory [8], SMART [9], Eon [10], and the Asgaard Project [11].

#### 4 *PROforma* – a task based approach to guideline representation

The term “Clinical Guidelines” describes a heterogeneous range of textual material designed to support practitioners in their decision-making. Individual guidelines may describe interventions concerning populations or individuals. These interventions may be simple “atomic” actions or complex plans of therapy to be conducted over time. The decisions they describe include diagnostic, prognostic, therapeutic and risk assessment decisions. Unlike clinical protocols, which are intended to be followed rigidly, usually within special contexts such as clinical trials, clinical guidelines often outline general principles of management, to be considered by experienced staff as they exercise their clinical judgement. Guidelines therefore frequently contain ancillary information such as the evidence and literature on which their recommendations are based, guideline authorship, and the statements of the applicability of the material presented to particular patient groups.

When work began on *PROforma* in the early 1990s, the main requirements for the language were that it should:

- Be sufficiently expressive to fully represent a range of clinical processes
- Be sufficiently general to describe processes in any clinical specialty, from routine care to clinical research
- Use concepts that are intuitive for clinical users

whilst ensuring that

- Processes specified in the language can be enacted by machine
- The semantics of the language are demonstrably sound
- Applications can be automatically checked for consistency and other properties

*PROforma* combines the features of a formal specification language as developed in software engineering with the features of knowledge representation languages as developed in artificial intelligence [12]. The *PROforma* language structure is based on a simple but versatile clinical process model referred to as the *domino model*. This model was abstracted from a variety of empirical studies of clinical decision-making and the development of aids to support patient management. The logical form of the domino model is shown in figure 1.

As described earlier, decision-making is a core function of any guideline representation and enactment schema. Statistical decision theory is the most formal decision model but in many respects has proved difficult to use and unattractive to clinicians. Decision theory based on mathematical probabilities is sound and well understood, but decision support systems based on statistical decision theory are often inflexible and difficult to use, and of limited applicability to knowledge-rich domains such as healthcare where precise parameters like quantitative probabilities are seldom known.

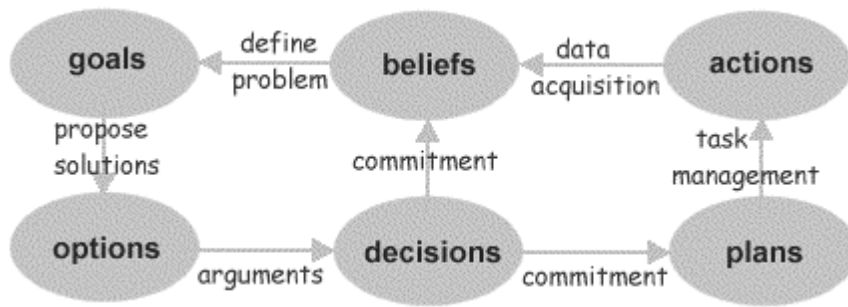


Figure 1: The domino process model. The left hand side of the diagram represents decision-making, the right represents plan enactment. Given a set of beliefs, an agent may infer goals and various solutions to these goals. If there are multiple options, such as alternative diagnoses or treatments, the agent must consider the arguments for and against these alternatives and make decisions based on the validity of each of the arguments.

Conversely, knowledge based DSSs are more flexible, but have not been based on a well understood theory of decision making, with existing tools and notations not sufficiently rigorous for safety-critical applications. The RED (Rigorously Engineered Decisions) project set out to unite the strengths of both approaches, and led to the development of both the domino model and a formal framework underpinning the concepts described to in the model [13]. The domino model describes a relationship between actions, decisions, beliefs, plans, problem goals and candidate solutions and the inference and processes linking them.

A further result of the RED project was the reification of the domino model into a minimal set of executable generic tasks - enquiries, decisions, plans and actions (figure 2). Tasks are formal software objects that can be composed into networks representing clinical guidelines or other processes, and it is from these tasks, and the logical constructs associated with each task, that the PROforma language is derived.

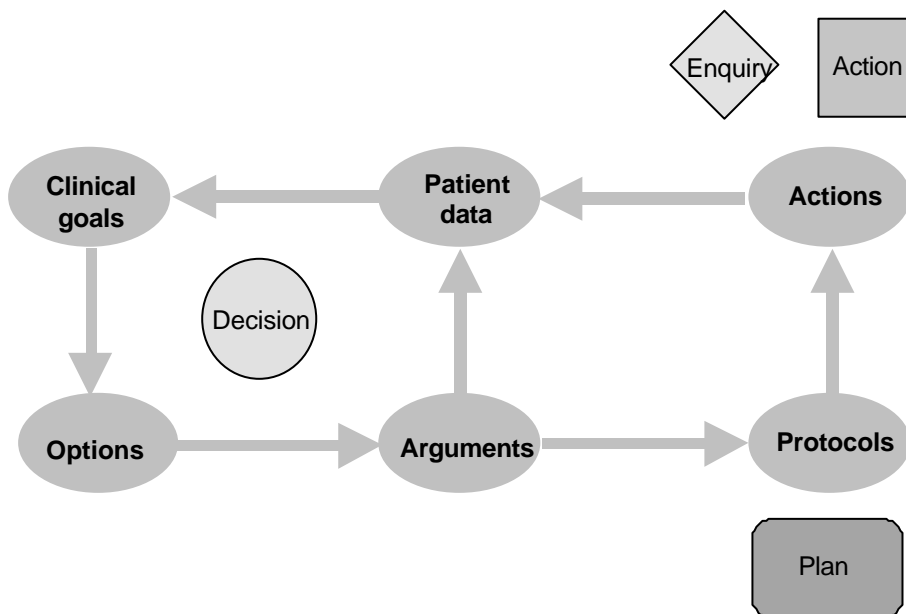


Figure 2: The relationship between the domino model of the clinical process and PROforma tasks derived from the model. Arrows represent logical inference and nodes represent data structures.

The main features of the *PROforma* language resulting from this approach are:

- A compositional and reusable knowledge representation
- Expression of clinical processes in terms of tasks, not rules
- The generation of knowledge components, not programs
- The use of ontologies as opposed to frames and objects
- A formal, declarative interchange format
- Sound foundations for representing and reasoning about uncertainty and time

*PROforma* may be distinguished from other schemas for clinical guideline representation by the abstract nature of the primitives it provides. As described above, clinical guidelines may refer to vast range concepts, such as diagnostic, prognostic or treatment decisions, or the gathering of data via laboratory tests, imaging or clinical examination, for example. Rather than attempt to explicitly describe all such concepts, it has been our aim to provide a minimal set of abstract terms, with properties that, we hope, allow a sufficient range specific instantiations to be defined. This has enabled us to define a clear behavioural semantics for the language, whilst retaining expressiveness and flexibility.

## 5 *PROforma* representation in software

*PROforma* technology can be viewed both as a kind of *logic programming* system, in that it supports inference in propositional and predicate logics, together with certain non-classical logics, and as an *object-oriented* system, in which the main objects are encapsulated procedures for achieving particular goals, which are referred to as “tasks”. A *PROforma* guideline consists of a set of tasks and a set of data items whose values are accessible to those tasks.

### 5.1 *The Task Ontology*

There are four classes of *PROforma* task: *Decisions* represent the choice points in a guideline, *Actions* represent some action performed by a human or software agent, *Enquiries* represent the acquisition of data from some external source, and *Plans* allow other tasks to be composed in order to perform some complex operation. Each task has a set of properties for which values may be defined. The set of properties depends on the class of the task, however certain properties are common to all tasks irrespective of class and hence may be described as belonging to a generic task class of which the four classes mentioned above are extensions.

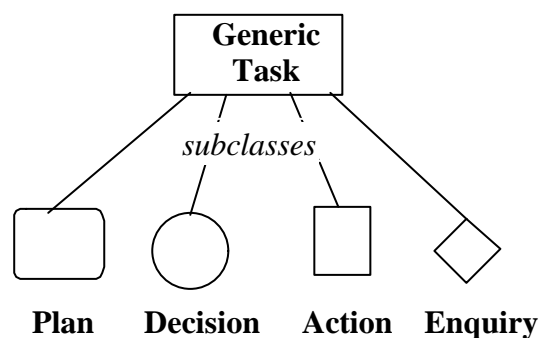


Figure 3 : *The PROforma task ontology*

### 5.1.1 Generic Task Properties

Certain properties are common to all tasks, others are specific to each class of task. All tasks may (optionally) have values for the following properties:

- Preconditions – these are logical conditions that must be true in order for the task to start.
- Postconditions – these are logical conditions that may be assumed to be true after the task has finished. Where necessary the task will alter the values of data items in order to ensure that its postconditions are true.
- Goal – a logical condition expressing the situation that the task is intended to bring about. A task will terminate as soon as its goal is achieved.
- Description – this documents what the task does and may refer to external sources of information justifying and explaining the operations described by a task or a guideline.
- Trigger – a message that may be passed to the task in order to start it even if its parent plan has not scheduled it to start (see section 5.1.3).

### 5.1.2 Decision Properties

In addition to the generic task properties, a decision may have values for:

- Candidates – these are the options between which the decision chooses. Each candidate is associated with a set of arguments. An argument is a logical condition with an associated weighting. A support value can be determined for each candidate by summing the weightings of those arguments that are true when the decision becomes active. Each candidate may also be associated with a recommendation rule, which determines whether it is considered advisable for that candidate to be chosen.
- Choice mode – may be either *single* (only one candidate may be chosen) or *multiple* (many candidates may be chosen).

### 5.1.3 Plan Properties

In addition to the generic task properties a plan has:

- Components – these are the tasks that the plan contains.
- Scheduling Constraints – define the order in which the component tasks are executed.
- Abort Conditions – logical conditions which, if true, will cause the plan to fail.
- Termination Conditions – logical conditions which, if true, will result in the plan terminating even if some of its candidates have not yet been considered for execution.

### 5.1.4 Enquiry Properties

In addition to its generic properties the definition of an enquiry includes a set of *sources*. These are the names of the data items for which values are to be requested. Each source may (optionally) be associated with a query that defines an external database and an SQL query that will yield the requested values. In the absence of a query the means by which

data values are obtained is not defined in the guideline and hence some external human or software agent must take appropriate action.

### 5.1.5 Action Properties

As well as its generic properties, the definition of an action includes the action's *procedure*. This is either a free text string defining some operation to be performed by a human agent, or an SQL statement describing a database update, along with a URL identifying the database on which that update is to be performed.

## 5.2 Computer Aided Software Engineering

In addition to the definition of the language structure, its syntax and the behavioural semantics of its enactment, the core technology of PROforma also embodies a number of software tools and a set of APIs. The PROforma language is intended for use in building practical systems for clinical task management. A PROforma technology will typically provide an authoring system for “proformalising” clinical protocols or other procedures in the language, and a software engine for enacting the tasks in the clinical setting such as the Arezzo® authoring and testing system, a PROforma technology developed by InferMed Ltd ([www.infermed.com](http://www.infermed.com)) (figure 4).

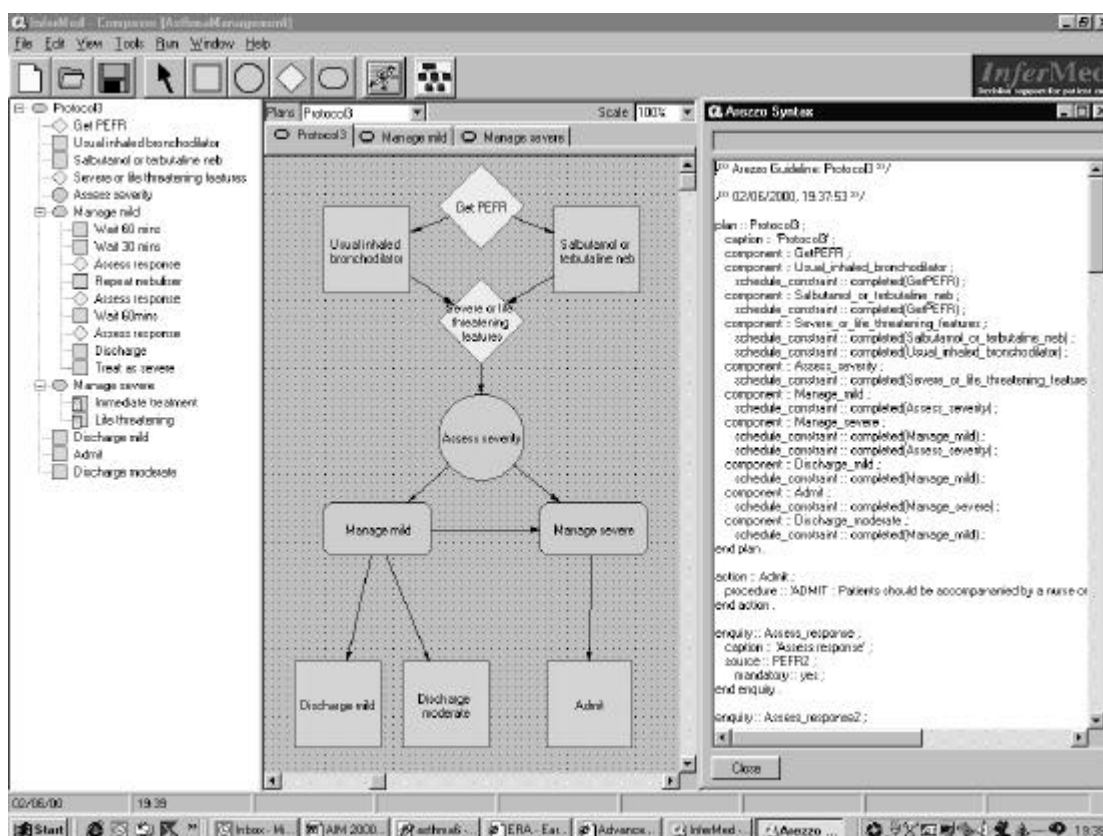


Figure 4 : The Composer system developed by InferMed for the graphical construction of task networks. The left hand panel presents an overview of the tasks that comprise the guidelines, the centre panel displays a graphical representation of the tasks and their scheduling constraints, and the right hand panel illustrates the PROforma guideline specification produced by the system. The software incorporates a automatic syntax checking, and a testing environment for validating the behavior of the guideline produced.

## 6 Experience of PROforma in use

The development of the PROforma language and associated technologies has enabled us to develop and evaluate a range of applications across a number of different clinical domains.

### 6.1 Support for drug prescribing

The CAPSULE system (Computer Aided Prescribing Using Logic Engineering) was developed to assist general practitioners (GPs) with prescribing decisions. CAPSULE analyses patient notes and constructs a list of relevant candidate medications, together with arguments for and against each (based on 9 different criteria, including efficacy, contraindications, drug interactions, side-effects, costs etc). A controlled study with Oxfordshire general practitioners showed the potential of CAPSULE for very substantial improvements in the quality of prescribing decisions [14]

### 6.2 Risk assessment in Primary Care

The RAGS (Risk Assessment and Genetic Screening) application is designed to assist clinicians in providing appropriate advice to patients who may be at risk of a hereditary predisposition to breast, ovarian and colon cancers and other diseases with a genetic component. RAGS assists in the construction of a family tree, and uses a PROforma task model to assess an individual's risk in the light of their family history and suggest whether or not the person should be referred for further investigation and/or counselling. Initial trials with GPs have produced very promising results [15]. In a comparison with Cyrillic, a leading commercial pedigree/risk assessment product, RAGs was chosen as the preferred risk assessment tool 91.7% of the time, as opposed to 5.6 % for Cyrillic and 2.7% for pen-and-paper techniques [16].

### 6.3 Image analysis and understanding

The CADMIUM (Computer Aided Decision Making and Image Understanding in Medicine) project set out to address the problem of integrating *symbolic* representations of knowledge with images and similar analogue data from medical devices in the context of mammographic cancer screening [17]. Several imaging protocols have been developed in an effort to investigate the technical and theoretical issues in *proformalising* image understanding functions. We are now focussing on integration of image processing and understanding functions with decision support within the context of a clinical trial studying the use of Tamoxifen.

### 6.4 Chronic disease assessment and treatment

Studies are currently being undertaken to evaluate the Arno system for the diagnosis, investigation and management of chronic pain syndromes. This complex system was developed by InferMed Ltd., using their Arezzo® authoring and enactment tools for the PROforma language. Arno offers support in relating a patient's symptoms to their probable causes, as well as advising on appropriate analgesia and adjuvant therapy. The Arno specification includes 32 plans, 19 actions, 29 enquiries covering over 200 individual items of information, and 140 decisions which embody over 1000 argument rules (see [www.infermed.com](http://www.infermed.com)). It is the largest PROforma application developed to date and shows that the technology has practical scaling properties.

### 6.5 Hospital Care

We have also developed number of applications for use in hospital settings. RetroGram™ has been developed by InferMed to assist clinicians in choosing appropriate anti-retroviral

therapy for patients with HIV, based on the individual pattern of genotypic mutations present in a patient specific HIV strain. The use of RetroGram™ in clinical practice is being evaluated as part of the Genotypic Resistance Evaluation to Aid Therapy-switching (GREAT) trial.

A decision support system to assist with prescribing and scheduling decisions in the treatment of childhood acute lymphoblastic leukaemia (ALL) is being developed within the context of the Medical Research Council's ALL '97 trial. Significant requirements of this project include the integration of the PROforma enactment engine with conventional clinical-trial databases and electronic patient records, as well as decision functions for supporting continuity of care as patients move between geographically separate treatment sites.

We believe that a generic solution to the problem of integrating decision support functions with electronic patient records is an urgent requirement for many practical decision support and workflow applications. This problem is being addressed within the context of the SyNeX project, which is developing an approach to the implementation of patient records based on a federation of independent databases and patient data sources. A demonstration interface between the SyNeX patient record system and a PROforma anti-coagulation therapy protocol is currently being developed.

## **7 Ongoing challenges for PROforma**

A number of recurrent themes have emerged in the development and testing of the applications described above. In general, difficulties have concerned the integration of applications into the local computing environment and the workflow of clinical consultation and process, rather than the ability to adequately describe a particular clinical process or concept in the PROforma notation.

### *7.1 Expressiveness*

Unlike a conventional logic language, such as Prolog, PROforma supports high-level objects (e.g. notably tasks but also general object hierarchies such as ontologies of diseases, drugs etc.). PROforma also makes use of non-classical logics (e.g. temporal and argumentation logics) and “enactment” rules, which implement a form of meta-interpretation over declarative descriptions of task properties (e.g. task goals, scheduling and temporal constraints, plus termination and abort conditions on plans). The language is very expressive for formalising task-oriented procedures though inevitably though this is at some cost in general logic programming capabilities.

### *7.2 Readability and intuitiveness of the language*

A PROforma task specification is intended to be as easy to understand as possible, ideally by domain-specialists who are not programmers, by adopting familiar, intuitive and modular constructs as the basic elements of the language (e.g. beliefs, goals, decisions, plans, actions etc). These are expressed in a simple and uncluttered syntax and the intended behaviour can be directly appreciated from task enactment diagrams represented with straightforward graphical conventions (figure 5). It is not a requirement that domain specialists should be able to *construct* such specifications, however, since a sound specification will frequently depend on an appreciation of programming disciplines.

### 7.3 Formally sound foundations

So far as is possible the concepts of *PROforma* are grounded in a well-defined and formally verified logical framework. The logical theory underlying the current *PROforma* task specification language is closely related to the RED formalism as described in depth by Fox and Das [18]. However, there are some differences between RED and *PROforma* because of the stronger object-oriented style of the latter, notably the introduction of task classes and inherited properties over these classes. A demonstration of formal soundness has been completed for the RED formalism but the equivalent for *PROforma* is an obligation for future work.

### 7.4 Infrastructure and integration

Two *PROforma* authoring and enactment systems have been developed to date, *InferMed*'s Arezzo toolset and a new generation of web-oriented tools. *InferMed*'s MACRO clinical trials manager uses a *PROforma* enactment engine for scheduling the presentation of electronic case report forms, checking validity of data entries, and decision support during the trial. The more recent engine is designed to support a client-server model, in which a *PROforma* guideline server can support parallel sessions and multiple guidelines accessed through browser-based user interfaces. Both authoring systems are single-user, standalone guideline editing environments [18] [d1] and both enactment engines are capable of being embedded in host applications, such as patient record systems and clinical trial systems.

The Solo client system can also be used to link a guideline running on a client machine to other local applications, such as the image-processing and genetic analysis applications described above. To support integration and communication with other information systems, we have recently developed an XML document type definition (DTD) describing the *PROforma* task set.

To date, much of our work has been based in environments such as primary care and out-patients clinics where the logistics of the interaction between clinician, patient and computer are relatively straight-forward. Hospital environments are considerably more complex, with many clinicians collaborating on the care of multiple patients over extended time periods. Wards with 30 or more patients may have perhaps only 1 or 2 computer workstations, inaccessible from the bedside. Mobile technologies will thus be essential in making decision support tools available as close to the locus of care as possible. We are exploring the role of such technologies in the ALL project described above. A general solution may well require handheld or other portable devices acting as 'thin clients' linked via a wireless LAN to a server hosting the *PROforma* engine and patient database. In the ALL project itself however, the limited number of clinicians interacting with a given patient in a given time period makes running the *PROforma* engine and a lightweight patient record on the portable device itself a more pragmatically attractive approach.

## 8 The *PROforma* 2000 project

We are currently defining requirements for an updated specification of the *PROforma* language and technologies, collectively referred to here as *PROforma* 2000, drawing on the experience gained to date on these and other applications.

### 8.1 Extensibility

If *PROforma* becomes more widely adopted, guideline developers are likely to want to extend the current set of generic tasks: decisions, plans, actions and enquiries. As users

recognise general patterns in the applications they create they will wish to express these generalisations through the addition of new classes to the PROforma task hierarchy. It is currently possible to develop specialised versions of tasks (e.g. prescribing or diagnosis decisions) by populating the generic task structure. It is also possible to create task “templates” consisting of unpopulated task sets, and saving them in a task library for subsequent reuse (figure 5).

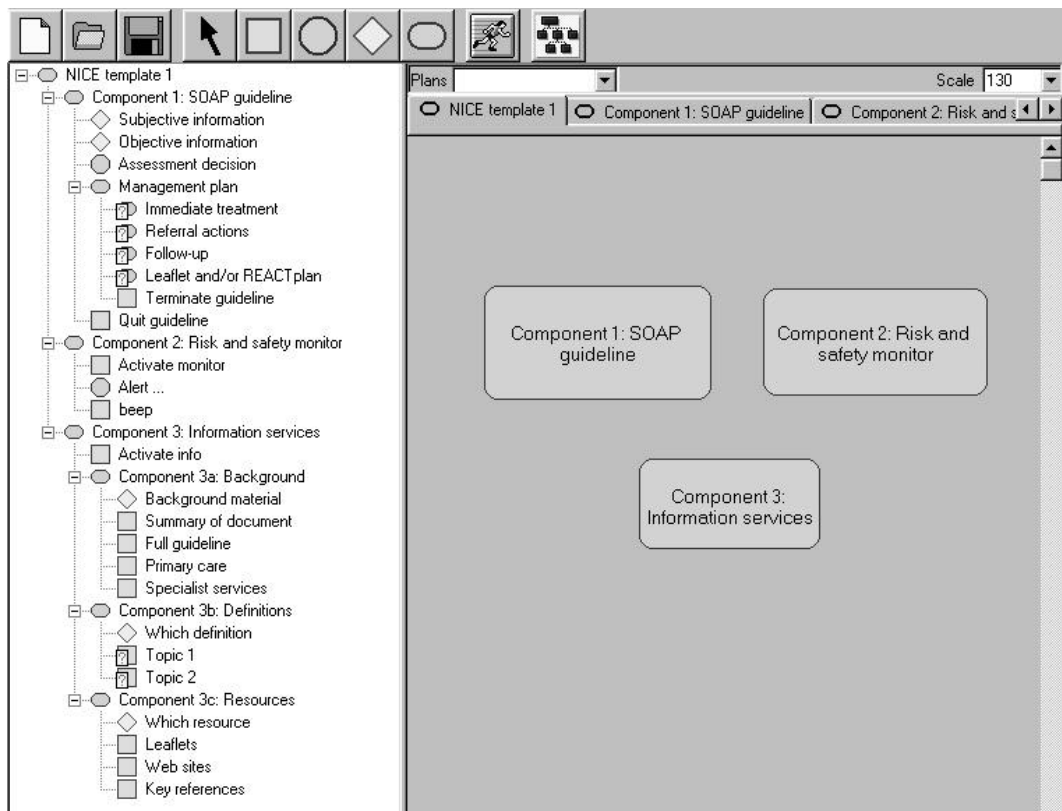


Figure 5: A guideline template for primary-secondary clinical referrals, based on text guidelines published by the UK National Institute of Clinical Excellence. The template has 3 components, each made up of a set of standard PROforma tasks, one is the referral guideline itself, one a simple monitor for detecting urgent or hazardous conditions, and one which can be populated with various kinds of conventional information services.

However there is only limited support for template creation and for the creation of entirely new classes of task (e.g. therapy design or data analysis tasks). Since such extensions will arise as a result of experience with the language and cannot be anticipated in advance it is important that the task hierarchy should be readily extensible.

We envisage that development of PROforma technologies will be an ongoing and process with continual input from guideline designers and clinicians. It is therefore important that designers should be able to create, distribute, share and experiment with extensions to the task hierarchy, function libraries and application viewers supported by PROforma 2000 and its associated development and enactment tools.

The need for extensibility has implications for the syntax and operational semantics of PROforma 2000 as well as for the interface between PROforma and software written in other languages. We also aim to specify the operational semantics of PROforma and its programming language interfaces in such a way as to allow new task classes and functions to be written in programming languages such as Java or C/C++ and distributed as modular components.

## 8.2 *Agent Communication*

The agent based computing paradigm is an increasingly influential approach to engineering of complex distributed systems. One feature of this paradigm is the recognition that interactions between different systems may not follow a simple client-server pattern. We are exploring ways in which *PROforma* 2000 might model interacting systems as separate agents pursuing their own goals, with the interaction between these systems treated as ‘conversations’ in which conflicts of interest are negotiated, ambiguities resolved, and common objectives identified. A number of languages have been defined for inter-agent conversation. Prominent among these are KQML and FIPA ACL [19], both of which classify messages in terms of *speech acts* which represent the intentions and beliefs of the sender of the message.

An important objective of the *PROforma* 2000 project is to integrate agent communication into *PROforma* in a manner that does not compromise its declarative semantics. One approach being explored is to extend the task hierarchy to provide specialised sender classes, which send messages to other agents, and receptor classes, which are triggered by the arrival of messages from other agents.

## 8.3 *Goal-based control*

A common objection to clinical guidelines, whether machine-readable or plain text, is that they express plans as series of tasks that are followed without reference to the intended clinical goal. This model makes it difficult to define strategies for recovery if a particular plan fails to achieve its objective. *PROforma* 2000 will address this problem by defining a syntax based on an explicit goal formalism, such as the task/focus model [20] in which the goal of a task is expressed as a kind of verb phrase (e.g. diagnosis of joint pain, treatment of headache, control of blood sugar). Goals are also important for controlling task termination (e.g. a plan should terminate when its goal has been achieved irrespective of whether all the scheduled tasks have been completed) and in plan replacement and plan repair (for a plan that has failed).

## 8.4 *Access to external knowledge representation systems*

The current *PROforma* language allows enquiries and actions to access external sources of knowledge such as databases. However in order to gain such access designers must incorporate site-specific details into guidelines thus making them difficult transfer between institutions. *PROforma* 2000 should allow interaction with external knowledge sources in ways that do not rely on ad hoc implementation details or site-specific information about the internal structure of such sources. The interaction should allow the guideline either to conduct conversations with external sources in some agent communication language or to incorporate such sources as a virtual or physical extension of its own knowledge base.

## 8.5 *Parameterisation of Tasks*

Variables in *PROforma* expressions may currently be instantiated to objects in the local knowledge base but only retain their bindings within the scope of the expressions in which they occur. *PROforma* 2000 will allow variables to retain their bindings throughout a task or plan and to be used to parameterise the effects those tasks have on the guideline’s local knowledge base or its actions on the outside world.

## 8.6 *Safety management*

The operational semantics and task hierarchy of *PROforma* 2000 will explicitly take into account the necessity of ensuring safe execution of guidelines and timely detection and rectification of hazardous conditions by incorporating tasks whose role is to ensure the

safety of the plans of which they form a part. Fox and Das [18] propose the use of specialised safety agents (“guardian agents”) to monitor clinical situations, trends, decisions and actions in order to detect and act on possible hazards during task enactment. Fox and Das show how *PROforma* technology can implement this concept though the current version of the language, but it does not provide explicit constructs to support the idea. It is intended that *PROforma 2000* will provide such support, in the form of explicit safety goals and generic hazard management tasks and strategies.

## **9 Discussion and conclusions**

We believe that the task based paradigm represents a promising solution to the problem of developing guideline representation formats that maintain a declarative knowledge base whilst reflecting the constraints of the clinical workflow and process within which that knowledge is to be applied. We have found the particular task model adopted in *PROforma* to be both expressive and intuitive, with its simplicity and formal foundations being significant benefits.

Although the development of the *PROforma* task model appears to have been productive, we must now address a second set of challenges. The technical challenges of providing interfaces between decision support systems and other clinical information systems such as electronic patient records and laboratory systems are well known. Within the context of the *PROforma 2000* project, the definition of a clear API for *PROforma* technologies will be a first step towards supporting communication between *PROforma* and other systems. We also hope that the adoption of techniques from the agent-based paradigm will help to address the difficulties of communication and negotiation between guidelines, and facilitate the development of clearer behavioural semantics for goal-based guideline representation.

The development of more intelligent functions, such as plan repair and the dynamic modification of plans represents a longer term objective for the *PROforma* format. These features will require a deeper understanding of general cognitive functions and how these can be modelled in formal systems. A further challenge in providing decision support at the point of care is in integrating computer systems into the dynamic of the clinical consultation. We are exploring the notion of “use models”, in which we consider the impact of a computer on the complex discourse between patient and clinician. Introducing a computer as a “third party” in this discourse may have significant effects on the psychological aspects of the consultation, both for clinicians and patients. If the decision support systems we wish to provide are to be integrated into clinical practice, their design must support the subtleties of the human relationship that exists between patients and their clinicians so that they may enhance, rather than detract from, this relationship.

## **10 Acknowledgements**

We thank Richard Thompson, Ali Rahmanzadeh, and Christian Blum for their help in preparation of this manuscript.

## References

- [1] Kohn, L., Corrigan, J., Donaldson, M. (Eds). To Err is Human: Building a Safer Health System. National Academy Press, Washington, D. C., 1999.
- [2] Shiffman RN, Brandt CA, Liaw Y, Corb GJ, 1999. A design model for computer-based guideline implementation based on information management services. *J Am Med Inform Assoc* 1999; 6(2):99-103
- [3] Wraith SM, Aikins JS, Buchanan BG, Clancey WJ, Davis R, Fagan LM, Hannigan JF, Scott AC, Shortliffe EH, van Melle WJ, Yu VL, Axline SG, Cohen SN. Computerized consultation system for selection of antimicrobial therapy. *Am J Hosp Pharm* 1976; 33(12):1304-8
- [4] Shortliffe EH. Update on ONCOCIN: a chemotherapy advisor for clinical oncology. *Med. Inform.* 1986; 11(1):19-21
- [5] Shwe M, Sujansky W, Middleton B. Reuse of knowledge represented in the Arden syntax. *Proc Annu Symp Comput Appl Med Care* 1992; 47-51
- [6] Hripcsak G, Wigertz OB, Kahn MG, Clayton PD, Pryor TA. ASTM E31.15 on health knowledge representation: the Arden Syntax. *Stud Health Technol Inform* 1993;6:105-12
- [7] Fox J, Johns N, Rahmazadeh A, Thompson R. *PROforma*: A method and language for specifying clinical guidelines and protocols. In: J Brender, J P Christensen, J-R Scherrer P McNair (eds) *Medical Informatics Europe '96* IOS Press, Amsterdam 1996 pp. 516-520.
- [8] Ohno-Machado L, Gennari JH, Murphy SN, Jain NL, Tu SW, Oliver DE, Pattison-Gordon E, Greenes RA, Shortliffe EH, Barnett GO. The guideline interchange format: a model for representing guidelines. *J Am Med Inform Assoc* 1998; 5(4):357-72
- [9] Pisanelli DM, Consorti F, Meriardo P, SMART: a system supporting medical activities in real-time. *Stud Health Technol Inform* 1997; 43 Pt A:343-7
- [10] Tu SW, Musen MA. A Flexible Approach to Guideline Modeling. *AMIA 1999 Annual Symposium*, Washington D.C., 420-424. 1999.
- [11] Shahar Y, Miksch S, Johnson P. The Asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. *Artif Intell Med* 1998; 14(1-2):29-51
- [12] Fox J, Johns N, Rahmazadeh A. Disseminating medical knowledge: the *PROforma* approach. *Artif Intell in Med* 1998; 14(1-2):157-181
- [13] Das SK, Fox J, Elsdon D, Hammond, P. A flexible architecture for autonomous agents. *J. Experimental Theoretical Artif. Intell.* 1997; 9: 407-440
- [14] Walton RT, Gierl C, Yudkin P, Mstry H, Vessey MP, Fox J. Evaluation of computer support for prescribing (CAPSULE) using simulated cases. *BMJ* 1997;315:791-5
- [15] Emery J, Walton R., Coulson A, Glasspool DW, Ziebland S, Fox J. Computer support for recording and interpreting family histories of breast and ovarian cancer in primary care (RAGs): qualitative evaluation with simulated patients. *BMJ* 1999;319:32-36
- [16] Emery J, Walton R, Murphy M, Austoker J, Yudkin P, Chapman C, Coulson A, Glasspool D, Fox J. Computer support for interpreting family histories of breast and ovarian cancer in primary care: comparative study with simulated cases. *BMJ* 2000;321:28-32
- [17] Taylor P, Fox J, Todd-Pokropek A. The development and evaluation of CADMIUM: a prototype system to assist in the interpretation of mammograms *Medical Image Analysis*. 1999; 3(4), 321-337.
- [18] Fox J, Das S. *Safe and Sound: Artificial Intelligence in Hazardous Settings*. American Association for Artificial Intelligence and MIT Press, Cambridge: MIT, 2000
- [19] Labrou Y, Finin T, Peng Y. The current landscape of Agent Communication Languages. *Intelligent Systems*, 1999; 14(2):
- [20] Huang J, Fox J, Gordon C, Smale A. Symbolic Decision Support in Medical Care. *Artificial Intelligence in Medicine*. 1993; 5(5):415-430

Page:  
[d1]Check that this is the right reference!