Patterns: a simple but expressive data modelling formalism

Tony Austin* and Shanghua Sun

Helicon Health,
97 Tottenham Court Road,
London, W1T 4TP, UK
Email: tonyaustin@heliconhealth.co.uk
Email: postmaster.electronsea@gmail.com
*Corresponding author

Nathan Lea and Yin Su Lim

CHIME, University College London,
The Farr Institute of Health Informatics Research,
222 Euston Road,
London NW1 2DA, UK
Email: n.lea@ucl.ac.uk
Email: y.lim@ucl.ac.uk

Archana Tapuria

Department of Primary Care and Public Health Sciences,
King's College London,
3rd floor Addison House,
Guy's Hospital,
London, UK
Email: archana.tapuria@kcl.ac.uk

David Nguyen

Helicon Health,
97 Tottenham Court Road,
London W1T 4TP, UK
Email: davidnguyen@heliconhealth.co.uk

Dipak Kalra

CHIME, University College London,
The Farr Institute of Health Informatics Research,
222 Euston Road,
London NW1 2DA, UK
Email: dipak.kalra@eurorec.org

Abstract: The creation of a clinical application requires models that describe the structure of data in a way that can be displayed, exchanged and stored. A number of approaches for this have been proposed and are in widespread use. However, these are often complex and/or have shortcomings in the breadth of data that they are able to represent. The annotations facility provided by many computer languages could be used to include information shaping the development and run-time behaviour of a clinical application. If this were comprehensive, then annotations alone would be sufficient for conceptual modelling. A model for representing such annotations is presented and some examples shown and discussed. The paper concludes that such a formalism is simple to use while developing semantic concepts but is capable of representing information from many models simultaneously. It is well suited to the needs of clinical teams seeking consensus on the structure of records.

Keywords: database design; data models; conceptual modelling; electronic healthcare records; EHR structure; semantic interoperability; archetypes; annotations; model-driven development; MDD.

Reference to this paper should be made as follows: Austin, T., Sun, S., Lea, N., Lim, Y.S., Tapuria, A., Nguyen, D. and Kalra, D. (2016) 'Patterns: a simple but expressive data modelling formalism', *Int. J. Knowledge Engineering and Data Mining*, Vol. 4, No. 1, pp.74–92.

Biographical notes: Tony Austin began his career developing decision support systems for chronic disease management before gaining a PhD in the representation and curatorship of electronic healthcare records. He has combined these experiences in the validation of international standards and in the development of clinical applications for a variety of domains. Most recently, he is a founder and shareholder of Helicon Health Ltd., a company incorporated with the objective to reduce the incidence of stroke.

Shanghua Sun is a consultant specialised in the storage and analysis of data at scale, including working in healthcare on the structure and scalability of electronic healthcare records. Collaborating with multi-disciplinary teams, she has experience in data description and interoperability within academia and industry. She is currently transferring the patterns approach described in this paper to other fields in order to support ubiquitous and scalable computing generally.

Nathan Lea is a Senior Research Associate at the UCL Institute of Health Informatics (IHI) working on projects in clinical care and research. His research interests include the role of information systems in supporting healthcare delivery and empowering patients, and information governance in the use of genetic, health and social care records in clinical research, particularly in the age of information and big data.

Yin Su Lim graduated with a degree in Manufacturing from her native Malaysia and then a Masters in Computing from University College London in the UK. She next moved to the Centre for Health Informatics and Multiprofessional Education (CHIME) and began work as a full-stack developer specialising in the construction of research applications for semantic interoperability and clinical applications for healthcare. She is presently engaged in the development of a multi-centre pan-European application to capture and analyse stroke data.

Archana Tapuria is a Research Fellow at the division of Health and Social care at King's College London (KCL). She has been the clinical lead of a research team at UCL, on various health informatics projects and has been designing and developing clinical information systems. Her area of research is clinical knowledge modelling, clinical archetypes and EHR standards like EN13606. She has contributed to international best practice in clinical modelling for EHRs, through the EuroRec Institute and openEHR Foundation. She has been part of the 'EHR Technical Advisory Group' for the NHS CFH (Connecting for Health) clinical content.

David Nguyen graduated in Computer Science at University College London during 2006 and then became a Research Assistant in the area of Health Informatics at the Centre of Health Informatics and Multiprofessional Education (CHIME). After pioneering work in the development of healthcare data integrity and client applications he moved from research to industry with Helicon Health Ltd in 2012. He has recently moved again, to the publishing industry, and has been a Java Software Engineer for Elsevier Ltd. since May 2016.

Dipak Kalra is President of the European Institute for Innovation through Health Data and of the EuroRec Institute. He plays a leading international role in research and development of electronic health record interoperability standards, data and privacy protection and in the reuse of EHRs for research. He has led multiple European projects in these areas, including the IMI programme alongside pharma companies, hospitals and ICT companies. He is Clinical Professor of Health Informatics at University College London, Visiting Professor at the University of Gent, and a member of standards bodies including CEN, ISO and HL7.

1 Introduction

Although the creation of a clinical application is a significant engineering effort, much of this is not spent in the technical rendering of a mature clinical understanding into a computable form but instead on eliciting such an understanding in the first place. Successive European Electronic Healthcare Records (EHR) projects starting with the Good European Healthcare Record (Lloyd et al., 1995; Grimson et al., 1998; Dixon et al., 2001) have recognised that it is only possible to ratify the generic attributes of a record in advance, such as the date and time of recording and the author. Domain-specific information such as that a blood pressure contains a systolic and diastolic component, or that an APGAR birth assessment comprises five elements each with a score of 0, 1 or 2, requires downstream standardisation from practitioners or learned societies. Since clinical knowledge is constantly evolving any attempt to solidify it could not possibly be future-proof. For this reason, a so-called 'dual model' approach is adopted that separates generic attributes from those of specific clinical domains.

The approach is used most recently in the ISO standard EN 13606 for health record communication. Part 1 of this standard (ISO, 2008a) describes how a record should be structured when transmitted between systems and defers description of exactly what would be included in such a transmission to domain-specific models that need to be

defined by clinical communities. Part 2 of the standard (ISO, 2008b) sets out the requirements for a domain modelling formalism and provides a candidate object model and language that supports those requirements.

Dual modelling stands in contrast to a typical object-oriented approach where, for example, two parties to an exchange of cardiovascular data would expect a real 'BloodPressure' class from which instances would be directly derived. Not so here – an instance of the *constraint* model could restrict the infinite set of class names to one with that phrase but the transmitted instance of the *record* model, representing only authorship information directly, would convey just a pointer to the constraints against which it was validated.

A number of industry-standard formalisms could be used to support the dual-modelling approach. Some widely known examples are introduced in the remainder of this section.

1.1 Clinical terminology

Clinical models are not a replacement for, nor are they replaced by, terminologies, in a learning healthcare system (Delaney et al., 2012). The focus of a clinical model is in defining the mereological structure of a medical record including any containment or organisational hierarchy implied by a collection of values. A terminology focuses on providing semantic context for the values. For example, an 'asthma check' will usually comprise a 'peak expiratory flow rate' (PEFR) reading and a number of individual symptom details such as whether the patient has been waking at night. A terminology might describe the PEFR as being a respiratory reading so that a clinician can search for the latter and find the more refined PEFR alternative. But the terminology does not describe the containment relationship placing a PEFR within an asthma check. The structure of the asthma check is what the clinical model provides. It tells a recipient what data might be expected in a record and where, which makes it the defining point in a conversation between a recorder and a recipient.

The clinical model must therefore be able to attach codes from one or more terminologies or ontologies to concepts it describes (this is also known as 'terminology binding'). This enables a recipient to quickly take advantage of translations or additional semantic knowledge provided by those tools. A clinical model can also declare that the possible values for a field are taken from a subset defined by a terminology but care must be taken to ensure that the values are fully enumerated in the clinical model so as to avoid the danger that a recorder or recipient lacking the (possibly fee-based) terminology will be unable to create a value or interpret it. If an otherwise free-text value is coded in the record, the human-readable equivalent text must be stored along with the code for the same reason.

Usually, post-coordinated terms are not permitted by a clinical model. This is because the meaning of such terms can subvert the intended meaning of the enclosing structure. Consider a structure calling for a list of diagnoses that apply to a patient. Such a list would tend over time towards a complete statement of conditions a patient has or has previously had. However, armed with post-coordinated terms a clinician could add 'NOT asthma' to the list, clearly contrary to the intended meaning but nevertheless with no obvious terminology-independent validation possible that would prevent the addition.

1.2 Archetype definition language

'Archetypes' (openEHR Project, 2016) are a formalism originally proposed by the openEHR foundation, a not-for-profit charity setup by University College London (UCL) and Ocean Informatics PTY (representing academic and industrial partners) to foster collaboration among organisations interested in EHR development and use. Part 2 of the ISO EN 13606 standard (ISO, 2008b) is based substantially on the archetype definition language (ADL) formalism provided by the openEHR foundation at version 1.4 (Beale and Heard, 2007), the only significant difference being that the underlying object model rather than ADL is the basis of the standard. This leaves other formalisms free to exist so long as they can demonstrate that they are at least expressive enough to represent the same information model content.

The archetype is a generic constraint mechanism that can be used to constrain models in any domain of interest. Since archetypes describe constraints on objects and their attributes but makes no distinction between different underlying reference models, this is a powerful constraint methodology across computing generally. Unfortunately, the 13606 standard does not require such generality since the one and only model it requires constraining is the one given in the first part of the standard. Still, the complexity resulting from the broader generality of archetypes would still be reasonable if it could constrain other reference models at the same time (for example, if an HL7 model might also be constrained). However, closer scrutiny reveals that while the possibility of constraining multiple models is undoubtedly present as stated, they cannot be constrained simultaneously. This is because an archetype defines constraints *in terms of a reference model*. For example, the range of pressure on a blood vessel might be stated differently by clinical and physiological models. An archetype cannot apply two different constraints to the same clinical concept because it cannot differentiate the two uses of the same attribute. If a second attribute were defined, it would either have to muddy both reference models or be seen as introducing a new 'superset-model'. Converters between openEHR and ISO EN 13606 standard reference models are required, and exist, for example, from Valencia (Martínez-Costa et al., 2009).

The archetype is focussed purely on data validation, that is, how to ensure that record data submitted by a user or automated system meets criteria for validity such as being within numeric ranges or having a limited set of possible textual values. This is certainly a very difficult problem. However, it is not the only problem that a clinical modelling paradigm must solve. For a simple example, it is important that an interpreting clinician know that the systolic value of a blood pressure is always presented before the diastolic value. The archetype typically does not record such clinical knowledge. To do so would require another formalism (van der Linden et al., 2009). Although the object model forms the basis of the standard, most real archetypes are delivered in an ADL source file that is expected to be self-contained and have a fixed scope. It is possible to refer to definitions outside the local source using something called a 'slot' but more complex aggregation of content requires yet another formalism (called a 'template').

1.3 Semantic web

Key among the metadata standards outside the specifically healthcare environment are those from the World Wide Web Consortium relating to the semantic web (Berners-Lee et al., 2001). These are intended to address the challenges of moving from a

predominantly human-readable World Wide Web of hypertext markup language (HTML) pages to a more machine-comprehensible version.

Central to this is a simple formalism called the resource description framework (RDF) (Manola and Miller, 2004). This allows facts (represented by a predicate and a value) to be associated with a resource denoted by a uniform resource identifier (URI). For example, http://doi.org/someid title 'Patterns: A simple … etc'. A derivative, the web ontology language (OWL) (Hitzler et al., 2012), can describe ontologies at the level of detail needed for reasoning.

RDF demands relatively higher development complexity and discipline to enable machine-processability of data, and once annotated it is more vulnerable to automatic censorship and privacy violation. It is expected that facts about a resource will be crowd-sourced and by design RDF does support information issuing from more than one constraint model at the same time. However, this must be carefully curated as crowd-sourced data will in general be incomplete or conflict.

Meanwhile, the primary failing of HTML that gave rise to RDF has been addressed in other ways. Microformats (Microformats.org, 2005) and Microdata (Schema.org, 2015) are specifications for data that can be embedded in a regular HTML page to facilitate processing by machines.

In the healthcare space, all the models for records in widespread use are models of a document and therefore represented most appropriately using XML, not RDF, components. Of the World Wide Web Consortium standards that naturally suggests XML schema, not RDF or OWL, is the most obvious choice for an information model description.

1.4 Classfile annotations

A general purpose annotation facility was introduced to the Java™ programming language through JSR-175 in version 1.5 (JCP, 2004). Annotations are a relatively simple metadata facility that allows classes, methods and attributes to carry additional information that can be introspected at compile-time or run-time. Several languages share a similar facility. Annotations encourage a declarative style of programming that specifies what must be done, not how it must be done.

This has been widely used in the creation of frameworks and in particular, the Java Persistence API (JCP, 2009) uses annotations to describe how data can be persisted into a database. A second version of the record server built by the authors (Austin et al., 2011, 2013) took advantage of this to embed persistence information into classes from which the clinical data was derived. A simple example for alerts is shown in Figure 1. Here, the annotation '@Entity' tells the object-relational modeller (in this case, Hibernate™) that this is a class requiring persistence and the appropriate table model is defined with for example, the '@JoinColumn' and '@ManyToOne' annotations.

Since Hibernate is able to read classes annotated with information about how they should be persisted, it is of course no surprise that we can write frameworks that do the same, checking at run-time against data validation annotations (for example, the '@Minimum' of a 'PhysicalQuantity' data type) and laying out a form according to data presentation annotations (for example, observing the '@ViewOrder' of the elements in an ISO EN 13606 Entry class).

The real class package and class name, and the supported method, enact the 'contract' specified by the annotations and is boilerplate code that could be produced from them at

compile-time automatically. However, to do so would require further build effort for a generator that creates the automatic alternative. This is redundant given that the annotations already contain the knowledge needed. The modelling formalism described in this paper follows from this observation.

Figure 1 An annotated class (see online version for colours)

```
package record.clinical;
@ArchetypeIdentifier("Alert")
@ArchetypeName("Alert")
@DateLastVerified(1163158422000L)
@DateOfIncorporation(1163158422000L)
@Entity
@Entry
@LibraryName("record.clinical")
@PublicationStatus(PublicationStatusType.PRIVATE)
@SecondaryTable(name="Alert")
@Version(1)
public class Alert extends Entry {
  @ArchetypeIdentifier("Remarks")
  @Cardinality(value = CardinalityType.ONE_AND_ONE_ONLY)
  @JoinColumn(name = "Alert_Remarks_id", table="Alert")
  @ManyToOne(cascade = CascadeType.ALL, fetch = FetchType.EAGER)
  public Remarks getRemarks() {
    return (Remarks)super.getSingletonContent("Remarks");
  }
}
```

2 Methods

The word ‘pattern’ is used in regular expressions to describe the set of constraints on an input for it to be a valid example. We reuse the same term to describe the set of constraints on a concept. The methodology specifically describes constraints applicable to a semantic concept rather than to a reference model in order to retain a broad scope.

2.1 Pattern model

The model itself is extremely simple. A constraint has one or more arguments (for example, the constraint @PatternIdentifier has one argument called the ‘identifier’ of type string), and a pattern has a number of constraints, whose arguments are then instantiated for the pattern. Arguments may have only one of a limited range of types; Boolean, integer, double, timestamp, string, ordinal, pattern and arrays of these. Certain constraints are permitted to occur more than once in a pattern, and certain arguments are permitted to take a default value.

A concrete example pattern is shown in Figure 2. This particular pattern uses constraints established by the ISO EN 13606 model and these are shown with light

highlighting. The model defines some class-level building blocks including the Element class which is the container for all actual data values. This 'clinic code' concept will have the StringWithLanguage data type when it is used for a record extract.

However, applying the @EN13606 constraint also implies the need for curatorship information, and the presence of this can be checked at any time. The dates of incorporation and last verification are the dates of original authorship and last change respectively, there is an author noted, an (in this case) English string representing a description of the pattern, a basic library categorisation of the concept if a full terminological categorisation is unnecessary, an identifier and name, a status, and finally a version.

Finally, a constraint is specified on the length of the string (shown in darker highlighting). This is among several additional validations that match those available in XML schema.

Figure 2 An example value pattern (see online version for colours)

Clinic Code

```
@DateLastVerified(TIMESTAMP verification="2012-03-14")
@DateOfIncorporation(TIMESTAMP verification="2012-03-14")
@DefinitionProvidedBy(STRING author="Shanghua Sun, CHIME, UCL, UK")
@Description(STRING language="en_GB", STRING description="A code representing a specific clinic.")
@EN13606()
@Element()
@LibraryPath(STRING path="clinical.clinics")
@MaxLength(INTEGER length=255, BOOLEAN inclusive=true)
@PatternIdentifier(STRING identifier="ClinicCode")
@PatternName(STRING name="Clinic Code")
@PublicationStatus(ORDINAL publicationstatus=DRAFT)
@StringWithLanguage()
@Version(INTEGER version=1)
```

The 'pattern' argument type enables us to conveniently represent child nodes in a record hierarchy (see also Section 3.2). In ISO EN 13606, the top-level container is known as the 'composition' and contains the medico-legal information associated with commitment and revision of data. An example from a real application is shown in Figure 3.

Figure 3 An example container pattern (see online version for colours)

NHNN Clinics

```
@Composition()
@Content(PATTERN child=$119, STRING label=<DEFAULT>, INTEGER n=0, INTEGER m=1)
@DateLastVerified(TIMESTAMP verification="2012-12-18")
@DateOfIncorporation(TIMESTAMP verification="2012-03-14")
@DefinitionProvidedBy(STRING author="Archana Tapuria, CHIME, UCL, UK")
@Description(STRING language="en_GB", STRING description="National Hospital for Neurology and
Neurosurgery clinic attendances.")
@EN13606()
@GroupLabel(STRING distribution="CORTEXT", STRING heading="NHNN Clinics", INTEGER ordering=40)
@LibraryPath(STRING path="clinical.clinics.nhnn")
@PatternIdentifier(STRING identifier="NHNNClinics")
@PatternName(STRING name="NHNN Clinics")
@PublicationStatus(ORDINAL publicationstatus=DRAFT)
@Sensitivity(ORDINAL sensitivitylevel=CLINICAL_CARE)
@Version(INTEGER version=1)
```

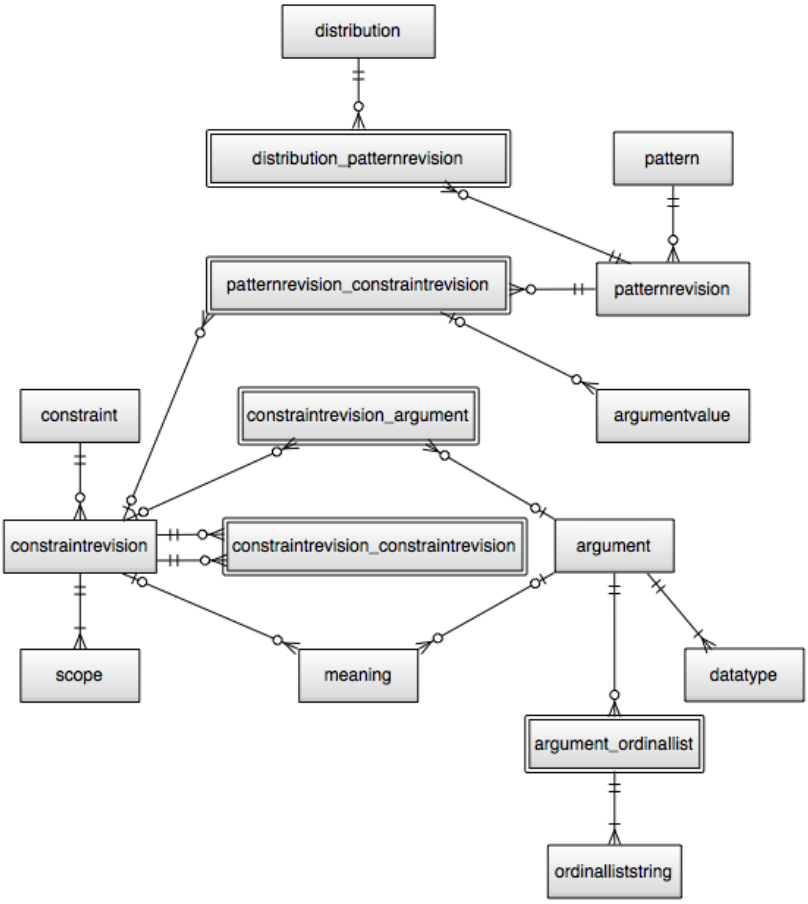
This composition has some @Content. A @Composition or any container may change the label for the child nodes it contains but in this case the @Composition is happy with the <DEFAULT> @PatternName of the contained Pattern.

The constraint list has also been extended with some presentation features. The dark @GroupLabel Constraint enables us to order panels in a navigation bar separately for each application the heading appears in.

2.2 アルチ (Aruchi)

Annotations are typically added to a program source file so that before and after completion the computational artefacts are real program code. However, Section 1.4 above notes that in fact the executable associated with the annotations is redundant for the purposes of describing the concept. In addition, although source files can be shared for editing via e-mail, or can have meaningful comments describing an edit attached in a source code repository, in comparison to a dedicated editor using source files discourages quick update or commenting and demands greater commitment from collaborators who are expected to first become ‘committers’.

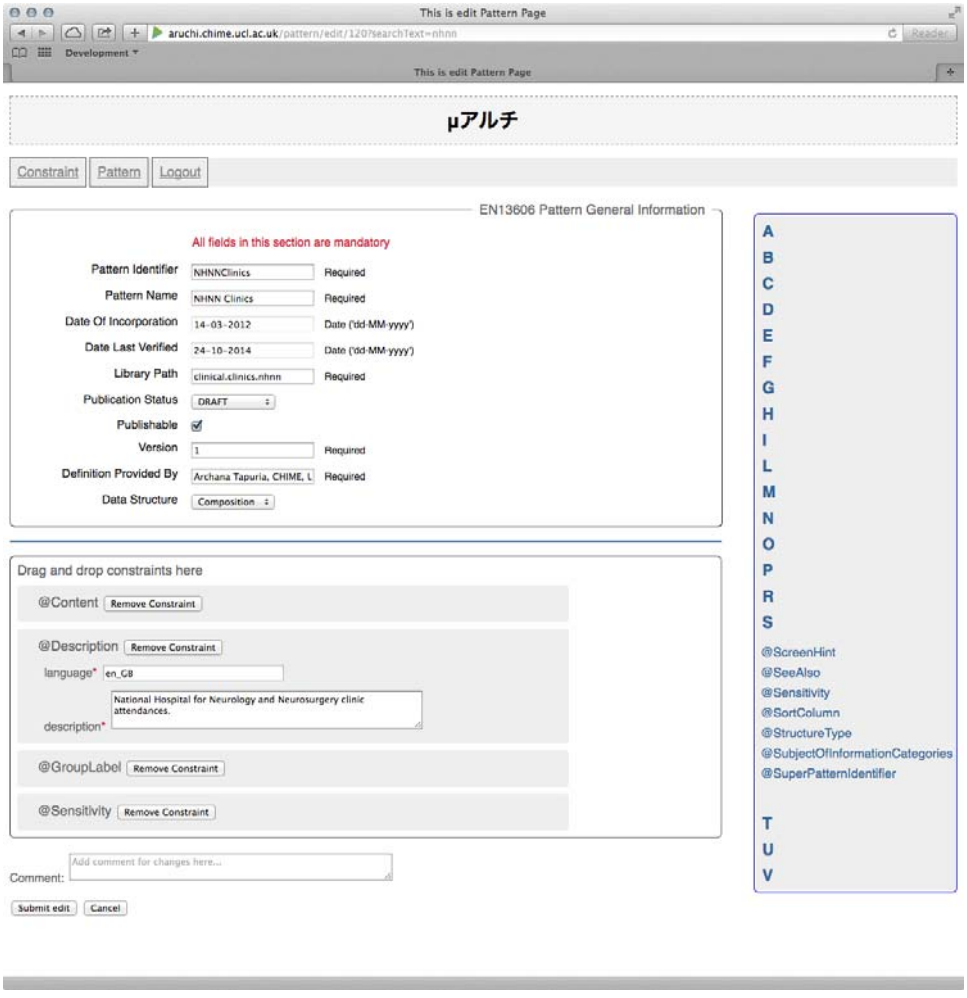
Figure 4 The relational structure of a pattern database



The simplicity of the pattern model lends itself in contrast to ready representation in a variety of underlying databases. A specifically relational model that satisfies the requirements for patterns is shown represented as an entity-relationship diagram in Figure 4. The actual creation of the database structure in the open-source PostgreSQL database is performed using scripts that make the process easy to replicate at other locations. These can then maintain their own separate repository of patterns.

Having an easily replicated relational model to store patterns significantly lowers the barrier to entry for organisations hoping to expose the implicit structure models in their existing applications. These models can include not only the types and aggregate structure of data entry screens but also human-readable descriptions of behaviours that populate fields or disable editing according to certain rules. By so doing, the patterns approach maximises the shared understanding of how data is represented and collected for clinical and research purposes.

Figure 5 Editing a pattern in Aruchi (see online version for colours)



Nevertheless, to encourage the adoption of these semantic resources by teams developing very different and often commercially competitive software, a still more inclusive approach is needed. To that end the development of a Web-based editor known as 'Aruchi' was begun (Lea et al., 2009). An example screenshot editing the NHNN Clinics example from Figure 3 is shown in Figure 5. This cloud-hosted tool is based on a single scalable instance of the model presented in Figure 4 and aims to act as an 'origin' repository enabling new applications and use cases to align themselves with semantic concepts already created. Consequently, it maximises the value of communicated records, exactly the purpose for which ISO EN 13606 was designed.

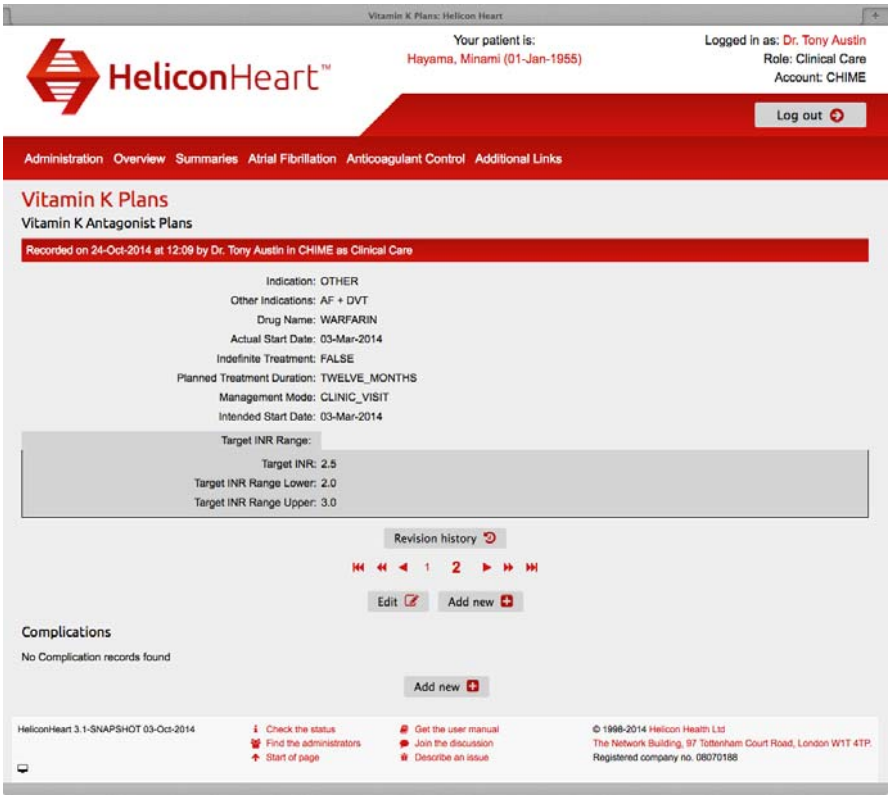
3 Results

The clinical models developed using the patterns method have underpinned both clinical and research activities. Two examples are presented in this section.

3.1 HeliconHeart™

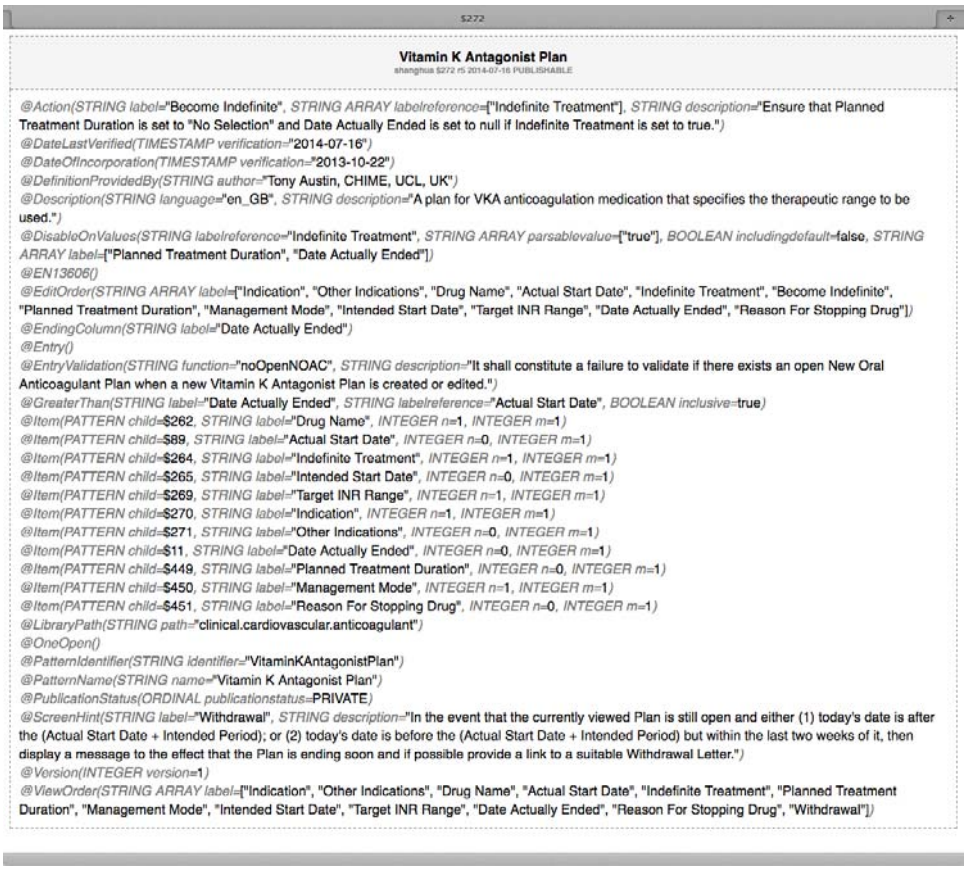
HeliconHeart™ is the flagship cardiovascular care application from Helicon Health Ltd., and is designed to support patients at risk of stroke or taking anticoagulant medication, including warfarin and the so-called 'new oral anticoagulants'.

Figure 6 A warfarin plan screen in HeliconHeart™ (see online version for colours)



Warfarin is a vitamin K antagonist that is potentially dangerous and can result in fatal complications. Patients taking warfarin have regular blood tests that determine the international normalised ratio (INR) of the blood, a measure of its propensity to clot. It has been shown that following a decision support guideline improves the time spent within an INR range that delivers therapeutic benefit (Rose, 2012). Using a computer-based decision support tool can improve time in range still further (Tapuria et al., 2013). The screen in Figure 6 shows the creation of a warfarin therapeutic plan which specifies the appropriate range for the patient to the built-in decision support system.

Figure 7 The model for the warfarin plan screen



The screen (and the database table that underpins it) is derived from the much more complicated pattern model shown in Figure 7. This also acts as a specification for extended features such as on-pane behaviours (for example, disabling the ‘planned treatment duration’ and ‘date actually ended’ fields if the clinician selects ‘indefinite treatment’) and hints (for example, pointing to a ‘withdrawal letter’ when the plan is about to end). Many jurisdictions demand that it be clear to a recipient what the context of an original recording was to ensure that interpretation can occur in a clinically safe way. By providing a comprehensive specification, it is completely clear both to an

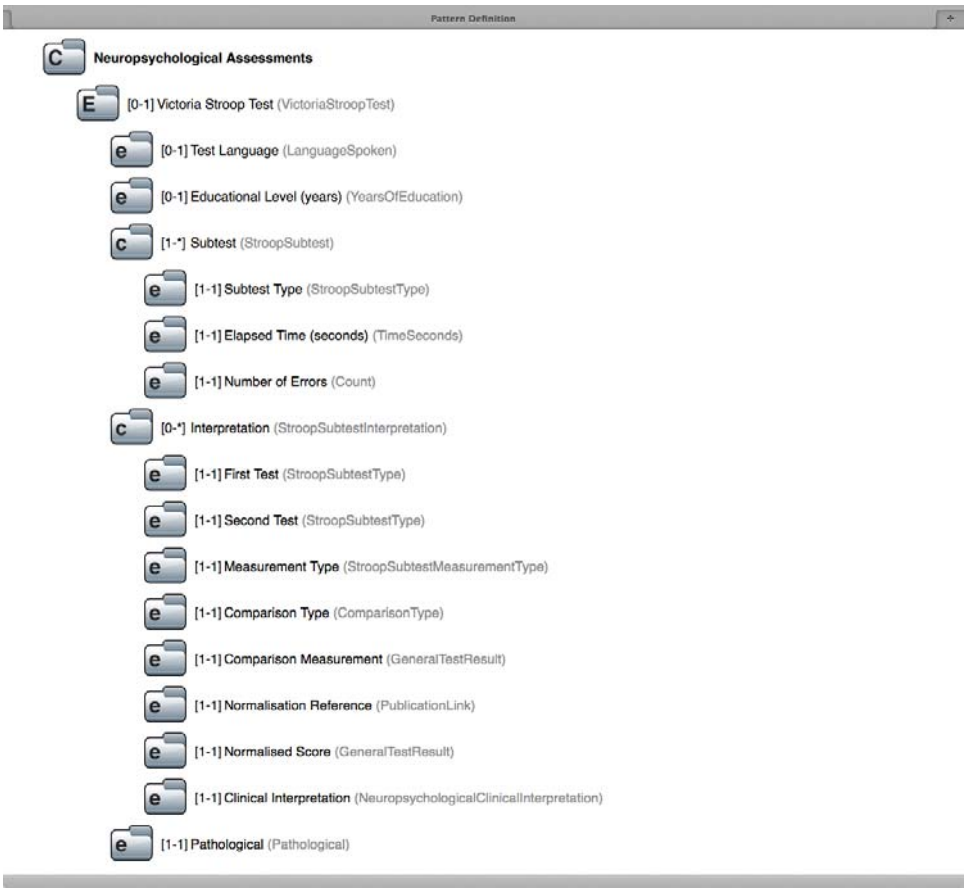
implementer what was intended by a feature and to a recipient what was available to an authoring clinician.

The application (Austin et al., 2015) underpins a distributed anticoagulation service (Austin et al., 2009) in North London that includes the catchment areas of five Clinical Commissioning Groups. The distributed service there includes not only hospital-based clinics but also General Practices and secondary prescribers, supporting over 100 users and having been used in the management of more than 15,000 patients. Helicon is now rolling out the software nationwide.

3.2 EMIF

The European medical information framework (EMIF project, 2016) is a 60 month, €56M project that aims to link up and facilitate access to diverse medical and research data sources. 'Big' datasets are fragmented across repositories and sources with different coding systems and natural languages, bound by different national and local legal and ethical constraints, and the actual stored information often does not have a semantic match across these.

Figure 8 The model for Victoria Stroop test



In an effort to present a single virtual repository that may be the target of research queries, it is first necessary to establish a single definition of the data that researchers may then use to frame their requests. Patterns have been applied to aid in this process (within the project they are termed 'EMIF knowledge objects').

An exemplar clinical area for the project is the diagnosis and management of dementia and Figure 8 shows an example model for a diagnosis instrument, the Victoria Stroop test (Bayard et al., 2011), that has been produced in collaboration with neurodegenerative disease practitioners. Here, the tree implied by the ISO EN 13606 container attributes is shown. The light grey bracketed identifiers are each fully modelled as patterns (not shown) and receive the labels presented in black when used in a container.

It is intended to incorporate this methodology in a large-scale pan-European federation of research repositories.

4 Discussion

The model for patterns is relatively simple at least in comparison to other widely-used formalisms, but it has required augmenting since the original concept was developed. It appears among similar tools in a recent evaluation by Moreno-Conde et al. (2016).

4.1 Constraints

Internal validation of argument values is in general not possible with this approach. Because the models being constrained are not part of the patterns representation itself, the latter has to assume that a knowledge engineer uses correct values. For example, an ISO EN 13606 content with a cardinality attribute '–1' cannot be declared unreasonable without also knowing what 'cardinality' is in the context of the EN 13606 model.

Nevertheless, it is at least possible to validate that constraints appear together where they should. For example, we know that all clinical models in an ISO EN 13606 environment must have the type composition, section, entry, cluster or element. It is therefore possible to ensure that if the @EN13606 constraint appears in a pattern, then @Composition or @Entry, etc. appear as well. The general approach is to include a table where constraint A can demand constraint C or suggest C. If neither of these are the case then the two constraints are unrelated. If A demands C then C must appear if A appears. If A suggests C and B suggests C then if C appears one of A or B must also appear.

This is a moderately powerful check although it can lead to confusing validation matrices. For example, the @Composition constraint both demands and suggests @EN13606 because @Composition requires the @EN13606 constraint to be present and @EN13606 requires one of @Composition, @Section, @Entry, @Cluster or @Element to be present.

Moreover, the naming and argument arrangement of constraints is still a matter of research. The example of warfarin plan in Figure 7 shows the balance the authors have tried to strike between specificity of constraint and human-readable description, but broader consensus may lean further towards one of these extremes. Our efforts have been predominantly directed at application design and build but the approach (with additional constraints) can at least be applied to security policy modelling (Lea, 2015) as well.

4.2 Comments

One of the key requirements for a tool that facilitates collaborative editing is storage of comments that allow users to justify their design decisions and propose new ones. However, this is not intended to describe the purpose of the pattern, the text for which is more properly stored in the @Description property of the pattern itself.

In our modelling work, we have noted with dismay more than one occasion where clinical domain experts forgot the reason for a previous request and asked for something different. The ability to explain requests as they occur is very valuable as, absent this facility, there is no basis on which to rebuff the new idea until what the previous choice made possible comes to light by its absence.

4.3 Distributions

A ‘version’ of a pattern refers to a single clinical concept even as it may collaboratively develop over many ‘revisions’. The former is represented as part of the pattern itself with the latter a part of the database model. For an example, consider a pattern for ‘emphasis’ which begins as a Boolean value indicating that a cohort of data was considered of higher importance by an author. The development of this pattern may go through many revisions, perhaps adding language translations or links to terminologies. However, it remains at version 1. Later, perhaps it is realised that a Boolean is insufficient to represent emphasis and the pattern is modified to declare an ordinal with several possible values. At this point, the collaboration must begin again around an updated concept. This would be considered a second version.

The process of collaboration between knowledge engineers and domain experts will tend towards a completed and usable pattern. However, not all stages of development have equal worth for the purpose of representing data. Even after a pattern is seen as usable, further development might diverge from broad acceptance before once again returning to a viable state. For this reason, each pattern may declare itself ‘publishable’ (sensible to use for data) and the database facilitates the creation of ‘distributions’ which are collections of publishable pattern revisions from which data may be created in a specific clinical domain. Multiple distributions may include the same pattern but only one version of a pattern may appear inside a single distribution. Distributions can use patterns at any point in their revision history, so long as they are publishable.

In fact, not only may individual patterns have less sensible iterations but so may distributions as a whole. The authors have found that clinical staff understand the implications of their modelling much more readily once it is rendered as an application they can use. Of course, these applications are only intermediate stepping stones on a path to an approved aggregation of patterns and have only limited applicability. A ‘snapshot’ distribution provides the content for applications regarded as temporary and not expected to retain real patient information. Meanwhile, distributions can also be ‘branched’ to allow more than one group to simultaneously explore different aggregations of patterns.

Since domain modelling forms a part of the ISO EN 13606 standard (part 2), it is sometimes assumed that semantic agreement between partners to an exchange would be dependent on there being a shared representation underpinning it. In contrast, a typical exchange partner who is as yet undecided about the technology to use for exchange, would see a full implementation of the standard as rather a burden when a simpler XML

schema could suffice. Looked at from the patterns perspective, a 'simple' XML schema would lack rigorous grounding in the custodianship of records (part 1) and agreement on the domain representation (part 2). Nevertheless, a ratified distribution could construct an XML schema that would 'look' simple but be fully informed on both counts.

4.4 Scalability

The approach works well in a small application development environment where interoperability is seen positively and the relatively higher cost associated with being thorough about the underlying models can be borne on that basis. However, it is not necessarily obvious that the approach would scale to a nationwide, continental or even global attempt to capture and inter-relate existing models. In particular, the scalability of the approach is limited by the rate at which appropriate constraints can be ratified, and the rate at which patterns can be created and subsequently found.

Section 2.2 above describes the drag-and-drop editor for patterns (Aruchi) that has been developed as part of this work. It emphasises a typical usage pattern where an example pattern of the sort required is found (for example, an 'integer with a minimum value') and then a new pattern is created using that as a template. This significantly improves the performance of knowledge engineers with large numbers of patterns to create, such as when beginning investigation of a new domain.

However, this may still not be quick enough for certain use cases. In clinical research, for example, it is more important to capture a description of the content of data than its complex curatorship information and governing value constraints. Instead, a 'rapid pattern development' environment can be envisaged where a minimal pattern model follows from a sketch of a data entry form that can be associated with the method used to extract data conforming to that structure.

4.5 Fragmentation

The formalism externalises and makes explicit a previously internal data model, in much the same way as a rule engine does for a knowledge base. The Aruchi tool is designed to facilitate collaborative development of the data model for the purposes of interoperability and exchange, but it is recognised that some (especially commercial) organisations will want to run bespoke instances of the server at least initially. The underlying model is deliberately simple at least in part to enable such servers to be established easily. However, to avoid fragmenting the pattern repositories and making them harder to search, there would be an advantage in having links to widely used ontologies so that searches could find patterns across server instances. A project already undertaken within the team (Panagiotou, 2011) has shown that the repository could offer patterns using an RDF syntax to establish links with broader ontological development.

5 Conclusions

The authors have shown that the patterns formalism provides a simple but powerful way of representing semantic concepts. The formalism is capable of including information from many models simultaneously and consequently it has broad applicability to the issues involved in clinical application design. It is also potentially useful in any other

situation where parties have not converged on a data model but instances can be exchanged at a coarser granularity. Two distinct modelling activities for two very different application areas have been discussed in this paper.

The formalism has a ready database representation and is thus well suited to collaborative editing. This lends itself well to the needs of clinical teams seeking consensus on the structure of records. Additional work could focus on rounding out such a collaborative editor for general use, which would have immediate benefits not only for application designers but in particular for interoperability specialists attempting to agree on the payload of a clinical exchange.

Acknowledgements

Part of this research was funded through the European Medical Information Framework Project (EMIF), Grant Agreement No. 115372, sponsored by the Innovative Medicines Initiative (IMI).

The authors would like to thank Helicon Health Ltd. for permission to use screenshots from their application in this paper.

References

Austin, T., Kalra, D., Lea, N., Patterson, D. and Ingram, D. (2009) 'Analysis of clinical record data for anticoagulation management within an EHR system', *The Open Medical Informatics Journal*, Vol. 3, pp.56–64.

Austin, T., Lim, Y.S., Nguyen, D. and Kalra, D. (2011) 'Design of an electronic healthcare record server based on part 1 of ISO EN 13606', *Journal of Healthcare Engineering*, Vol. 2, No. 2, pp.143–160.

Austin, T., Sun, S., Hassan, T. and Kalra, D. (2013) 'Evaluation of ISO EN 13606 as a result of its Implementation in XML', *Health Informatics Journal*; Vol. 19, No. 4, pp.264–280, DOI: 10.1177/1460458212473993.

Austin, T., Sun, S., Lim, Y., Nguyen, D., Lea, N., Tapuria, A. and Kalra, D. (2015) 'An electronic healthcare record server implemented in PostgreSQL', *Journal of Healthcare Engineering*, Vol. 6, No. 3, pp.325–344, DOI: 10.1260/2040-2295.6.3.325.

Bayard, S., Erkes, J. and Moroni, C. (2011) 'Victoria Stroop test: normative data in a sample group of older people and the study of their clinical applications in the assessment of inhibition in Alzheimer's disease', *Archives of Clinical Neuropsychology*, Vol. 26, No. 7, pp.653–661, DOI: 10.1093/arclin/acr053.

Beale, T. and Heard, S. (Eds.) (2007) *Archetype Definition Language (ADL 1.4.0)* [online] http://www.openehr.org/releases/1.0.1/architecture/am/adl.pdf (accessed 21 October 2016).

Berners-Lee, T., Hendler, J. and Lassila, O. (2001) 'The semantic web', *Scientific American*, May, pp.29–37.

Delaney, B.C., Peterson, K.A., Speedie, S., Taweel, A., Arvantis, T.N. and Hobbs, F.D. (2012) 'Envisioning a learning health care system: the electronic primary care research network, a case study', *Annals of Family Medicine*, Vol. 10, No. 1, pp.54–59.

Dixon, R., Grubb, P.A., Lloyd, D. and Kalra, D. (2001) 'Consolidated list of requirements', *EHCR Support Action Deliverable 1.4*, 59pp, European Commission DGXIII, Brussels [online] http://discovery.ucl.ac.uk/1603/1/R5.pdf (accessed 21 October 2016).

European Medical Information Framework (EMIF) Project (2016) Home [online] http://www.emif.eu (accessed 21 October 2016).

Grimson, J., Grimson, W., Berry, D., Stephens, G., Felton, E., Kalra, D., Toussaint, P. and Weier, O.W. (1998) 'A CORBA-based integration of distributed electronic healthcare records using the synapses approach', *IEEE Transactions on Information Technology in Biomedicine*, Vol. 2, No. 3, pp.124–138.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F. and Rudolph, S. (Eds.) (2012) *OWL 2 Web Ontology Language Primer*, 2nd ed. [online] http://www.w3.org/TR/owl2-primer/ (accessed 21 October 2016).

International Organization for Standardization (ISO) (2008a) *ISO EN 13606-1: Electronic Health Record Communication Part 1: Reference Model*, Geneva, Switzerland.

International Organization for Standardization (ISO) (2008b) *ISO EN 13606-2: Electronic Health Record Communication Part 2: Archetype Interchange Specification*, Geneva, Switzerland.

Java Community Process (JCP) (2004) *JSR-000175: A Metadata Facility for the JavaTM Programming Language* [online] https://jcp.org/aboutJava/communityprocess/final/jsr175/index.html (accessed 21 October 2016).

Java Community Process (JCP) (2009) *JSR-000317: JavaTM Persistence 2.0* [online] https://jcp.org/aboutJava/communityprocess/final/jsr317/index.html (accessed 21 October 2016).

Lea, N. (2015) *Design and Development of a Knowledge Modelling Approach to Govern the Use of Electronic Health Records for Research*, PhD thesis, University College London, London, UK.

Lea, N., Austin, T., Hailes, S. and Kalra, D. (2009) 'Expression of security policy in medical systems for electronic healthcare records', in *International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering*, Presented at the *World Academy of Science, Engineering and Technology*, Vol. 3, No. 5, pp.470–474, Tokyo, Japan.

Lloyd, D., Kalra, D., Beale, T., Maskens, A., Dixon, R., Ellis, J., Camplin, D., Grubb, P. and Ingram, D. (Eds.) (1995) 'The GEHR final architecture description', *The Good European Health Record Project: Deliverable 19*, European Commission, Brussels [online] http://www.webcitation.org/query?url=https%3A%2F%2Fwww.ucl.ac.uk%2Fchime%2Fresearch%2Fgehr%2Fdeliverable-19.pdf&date=2015-08-08 (accessed 21 October 2016).

Manola, F. and Miller, E. (Eds.) (2004) *RDF Primer* [online] http://www.w3.org/TR/2004/REC-rdf-primer-20040210/ (accessed 21 October 2016).

Martínez-Costa, C., Tortosa, M.M. and Fernández-Breis, J.T. (2009) 'Towards ISO 13606 and openEHR archetype-based semantic interoperability', in *MIE 2009*, DOI: 10.3233/978-1-60750-044-5-260.

Microformats.org (2005) *Specifications* [online] http://microformats.org/wiki/Main_Page#Specifications (accessed 21 October 2016).

Moreno-Conde, A., Austin, T., Moreno-Conde, J., Parra-Calderón, C.L. and Kalra, D. (2016) 'Evaluation of clinical information modeling tools', *Journal of the American Medical Informatics Association*, Vol. 23, No. 6, pp.1127–1135, DOI: 10.1093/jamia/ocw018.

openEHR Project (2016) *Archetype Technology Overview* [online] http://www.openehr.org/releases/AM/latest/docs/Overview/Overview.html (accessed 21 October 2016)

Panagiotou, S. (2011) *Clinical Pattern Representation in an RDF Format*, Unpublished MSc thesis, Centre for Health Informatics and Multiprofessional Education, University College London, London, UK.

Rose, A.J. (2012) 'Improving the management of warfarin may be easier than we think', *Circulation*, Vol. 126, No. 19, pp.2277–2279, DOI: 10.1161/CIRCULATIONAHA.112.141887.

Schema.org (2015) *Organization of Schemas* [online] http://schema.org/docs/schemas.html (accessed 21 October 2016).

Tapuria, A., Austin, T., Sun, S., Lea, N., Iliffe, S., Kalra, D. and Ingram, D. (2013) 'Clinical advantages of decision support tool for anticoagulation control', Presented at the *IEEE EMBS Special Topic Conference on Point-of-Care Healthcare Technologies*, Bangalore, India, 16–18 January, pp.331–334.

van der Linden, H., Austin, T. and Talmon, J. (2009) 'Generic screen representations for future-proof systems, is it possible? There is more to a GUI than meets the eye', *Computer Methods and Programs in Biomedicine*, Vol. 95, No. 3, pp.213–226.