Chapter 3
Towards Decentralised Clinical Decision Support Systems

Paolo Besana[1] and Adam Barker[2]

[1] Universite de Rennes, France
[2] University of St Andrews, United Kingdom

Abstract. The sheer quantity and complexity of medical information, even within a single speciality, is beyond the power of one person to comprehend. Clinical decision support (CDS) systems have been clearly demonstrated to improve practice by removing complexity and aiding the decision making process for clinicians. However, the specific pieces of information most relevant to a particular clinical decision are typically scattered over a wide range of databases, applications, journals and written notes. Centralisation of knowledge is becoming less practical as the volume and complexity of data increases. Through a motivating scenario taken from the field of cancer research, we argue against complete centralisation and towards an open, decentralised architecture, allowing domain experts to curate and maintain their own processes and data sets. We introduce the UK-based Safe and Sound project and propose an architecture based on PRO*forma*, a formal language for describing CDS systems and OpenKnowledge, an enabling technology for decentralised agent-based systems. We demonstrate that although more complex to initially model, our architecture scales with increasing complexity, is more flexible and reliable than architectures which rely on centralisation.

Keywords: Clinical decision support systems, medical guidelines, workflow, service choreography.

1 Introduction

Many science-based fields are facing a knowledge crisis, in that knowledge is expanding while economic resources and the human capacity to apply it remain finite. Medicine is perhaps the most prominent example; the field is the focus of a vast and productive research effort, but new tools and techniques are not always either quickly disseminated or effectively used. In the last thirty years, clinicians and computer scientists have developed technologies that could hold the key to managing that ever-growing complexity. Research on computerised Clinical Decision Support (CDS), Workflow and Knowledge Management systems has blossomed into a global movement, with applications in most areas of medicine.

Although healthcare brings enormous benefits to us all, there is compelling evidence that avoidable errors are common and patients are frequently harmed. Medical guidelines synthesise the latest evidence and best practice for a particular clinical

S. Brahnam & L.C. Jain (Eds.): Adv. Comput. Intell. Paradigms in Healthcare 5, SCI 326, pp. 27–44.
springerlink.com

condition or intervention. They are produced and used in many countries and have been shown to significantly improve healthcare outcomes [16]. CDS systems apply formalised (computer-interpretable) medical knowledge to patient data in order to arrive at patient-specific treatment recommendations and guidelines in a way that ensures best practice.

The evidence [18] that these technologies can have the desired effects (improving clinician performance, patient outcomes, and the cost-effectiveness of health centres, among others) is substantial, but at present no one knows how to deploy these systems in a safe, sound, and scalable way. A major challenge is how to integrate and deliver support for a variety of such services at the point of care, in the complex, unpredictable and ever-changing context of healthcare delivery, and how to do this in a flexible yet scalable and safe way.

In order to integrate software and data, academia and industry have gravitated towards service-oriented architectures. Service-oriented architectures are an architectural paradigm for building software applications from a number of loosely coupled distributed services.

Service oriented architectures are part of a wide family of middleware systems: every component exposes services accessible through the network. Complex systems can be pulled together, invoking services belonging to different and possibly external systems using workflow languages [4] like the Business Process Execution Language (BPEL) [2] or Yet Another Workflow Language (YAWL) [35]. These frameworks are based on a centralised, imperative paradigm: a central process controls everything, the other services are passive, unaware of being part of a larger, more complex workflow. In this approach, called *orchestration*, services are usually wired together into an application at design time, leaving little margin if, for example, one of the services is unavailable at run-time.

We claim that this approach does not scale well with the growing complexity of systems. This is the case, as we will see in the next sections, for medical procedures, where complex workflows, developed by committees, involve the interaction of many different actors such as desktop applications, web services, databases, diagnostic devices and monitoring tools and have to be adapted to the contingent and varying realities of hospitals and clinics. We advocate a different paradigm, based on *sharing choreographies* among actors. We believe that both design and execution of complex systems can benefit from this approach. At design-time, the choreography paradigm forces the developers to think in terms of the actors, their roles and their interactions, making them explicit and abstracting away from the details of their specific activities. While the general approach is to share description of services, and pull them together at design-time into a specific application, our approach takes the opposite direction and shares the choreographies. At execution-time, shared choreographies are the contracts that actors can search, verify and agree to follow: the binding with actors takes place at run-time, not at design-time.

The OpenKnowledge[1] project has allowed us to develop a distributed, open, peer-to-peer framework, focussed on shared interaction models that are executed by peers. In this paper we focus on the application of this framework to the

[1] `http://www.openk.org/`

coordination of medical guidelines, using as a case study the assessment procedure (called triple assessment) followed by a patient suspected of breast cancer.

This Chapter discusses the work of the UK-based Safe and Sound project[2] for eliciting grand challenges in ICT-driven healthcare and is structured as follows. In Section 2 we introduce the problem requirements of formalising and enacting medical guidelines, exemplified by the breast cancer assessment case study in Section 3. Then in Section 4 we discuss the advantages of a choreography-based approach to the problem, against an orchestration-based one. In Section 5.2 we cover OpenKnowledge, as a flexible, choreography-based framework that can address some of the requirements of medical guidelines. Related work is discussed in Section 6 and conclusions are presented in Section 7.

2 Medical Guidelines

Gaps between medical theory and clinical practice are consistently found in health service research. Care procedures can differ significantly between different health centres, with varying outcomes for patients, and medical errors could be avoided if standard procedures were followed consistently. One of the causes of discrepancies in care is the difficulty in distributing and sharing efficiently the large volume of information continuously produced by medical research. These issues have pushed the development of clinical practice guidelines: several studies [21] have shown that published guidelines can improve the quality of care.

Guidelines are usually defined by a committee of experts and are provided as booklets, often hundreds of pages long, covering relatively narrow fields or specific pathologies. Hundreds of guidelines are available, and generalist doctors are expected to be aware of, and follow, the guidelines relevant to each patient. The result is that guidelines are rarely followed, and inconsistencies in medical treatments are not reduced as much as hoped.

Information technology can improve the situation. Many clinical guidelines provide informal descriptions of workflows and rules that can be translated into formal, machine-executable, representations. Research has suggested that computerised clinical supports can improve practitioner compliance with guidelines [19].

Guidelines encode at least two separate types of knowledge: they specify the overall coordination between actors, both clinical and non-clinical (for example, the results of all the exams are sent to a multidisciplinary team for discussion and to the Electronic Health Record of the patient) and specific medical knowledge required in taking decisions (for example, dosing drugs or classifying a melanoma). Guidelines are developed for general adoption: the coordination specifications need to be adapted to the contingent realities of different institutions and clinics, and the medical knowledge need to be kept updated.

Finally, different guidelines specify medical procedures at different level of abstraction, from the general framework defining the milestones of screening, intervention and follow-up within the national health system, to the detail of the dosing of a specific drug. Depending on the condition and on the abstraction level, there can

[2] http://www.clinicalfutures.org.uk/

be varying degrees of freedom in the selection of the guideline, or in the selection of participating clinicians. For example, a woman can choose among several paths of care for her pregnancy, while the (more critical) procedure for a heart attack is more stringent. In order to enact the full pathway a patient has to go through the various specifications that should be integrated, selecting at each step the best one, and adapting them to the contingencies.

In the work described in this Chapter we use OpenKnowledge, a technology developed for distributed peer-to-peer systems, to represent, integrate and enact the coordination aspect of guidelines offering a level of flexibility that improves portability between different institutions.

Before the details of OpenKnowledge are discussed, we first introduce our case study and discuss the differences between the centralised and the distributed approaches.

3 Case Study: Assessment for Breast Cancer

We present as our case study the medical guideline for assessing the presence of breast cancer, because, as we will describe in more detail in the next section, a computer-based, centralised workflow has already been developed, making the comparison easier. This guideline is part of a larger more complex workflow, that includes periodical screening, surgical intervention and follow-up.

Breast cancer is the most commonly diagnosed cancer in women, accounting for about 30% of all such cancers. One in nine women will develop breast cancer at some point in their lives. In the UK, women with symptoms that raise suspicion of breast cancer are referred by their GP to designated breast clinics. To increase the accuracy of diagnosis, a combination of clinical examination, imaging and biopsy - known together as *triple assessment* - is recommended.

The first element of triple assessment consists of gathering the patient details and clinical examinations, and it is completed by a breast surgeon. If the clinical examination reveals an abnormality then the patient is referred to a radiologist for imaging (either ultrasound, mammography or both). If either the examination or imaging findings warrant it, then a biopsy is performed, either by a radiologist or a surgeon, and the tissue is sent to a pathologist for examination. The collective results from all three tests influence the final management of the patient. Depending on the resources available, different imaging and pathology laboratories might be selected every time the guideline is executed, possibly from different institutions.

A small number of 'worried well' patients may not qualify for either imaging or biopsy and could be discharged straight away. As the entire clinical process is distributed among three different disciplines and involves a number of different clinicians, a very close co-ordination and good communication between those involved is essential for the smooth running of the clinic.

4 Centralised and Distributed Models

The triple assessment model presented in [26] is designed according to a centralised principle, and the abstract workflow is shown in Figure 1. The centralised model

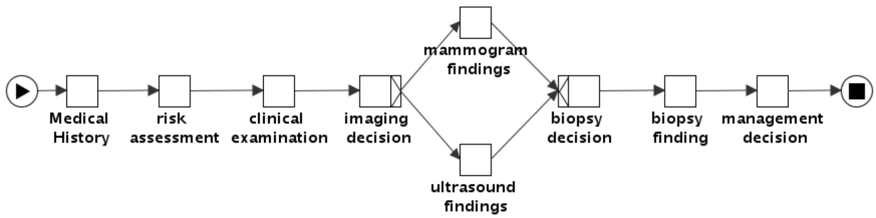

Fig. 1. Centralised representation of the triple assessment

has been implemented[3] in PRO*forma* [31], a process modelling language developed within Cancer Research UK. Together with the possibility of representing plans, sub-plans and actions, its strength lays in its decision support system, based on argumentation.

However, in PRO*forma* it is not possible to explicitly represent different roles in a guideline: roles are enforced by requiring different permissions to access the activities. Activities are queued in the todo list of a user, who is presented with this list after logging in. When the user finishes the activity, it is marked as completed and the workflow proceeds. While PRO*forma* provides a strong support for representing and reasoning about medical knowledge, it is currently inadequate for representing and executing coordinated activities between distributed actors.

The distributed nature of the procedure cannot be reconstructed from its representation as a centralised workflow. Moreover, medical knowledge specific to different participants, that is, the arguments and rules necessary for decisions in various phases, is centralised in a single procedure, making it impossible to reuse the same knowledge in different guidelines. If the model is implemented in another workflow language such as BPEL, the knowledge could be split amongst a group of services, possibly each based on PRO*forma*, but the process itself would still be represented from a single perspective.

A more realistic representation of the flow of the procedure is provided by the UML Activity diagram of Figure 2, in which the participants and their interactions are made explicit. Figure 3 is a corresponding UML Sequence diagram. There are five main actors: the breast surgery service (BSS), in charge of the first three activities in the workflow in Figure 1, the breast imaging service (BIS), responsible for the imaging decision and for the two alternative possible examinations, the breast pathology service (BPS), in charge of the biopsy, the multi-disciplinary team (MDT), responsible for the final decision together with the surgery service, and the patient.

The aim of this work is to allow the distribution of knowledge and information to the actors in charge, the reuse of knowledge in different guidelines, and the possibility of easily adapting guidelines to the realities of actual institutions and hospitals. We obtain this by separating the two different aspects of guidelines into two abstraction layers. The higher level consists in the choreography that specifies the expected

[3] A demo is available at: `http://tinyurl.com/tripleassessment`

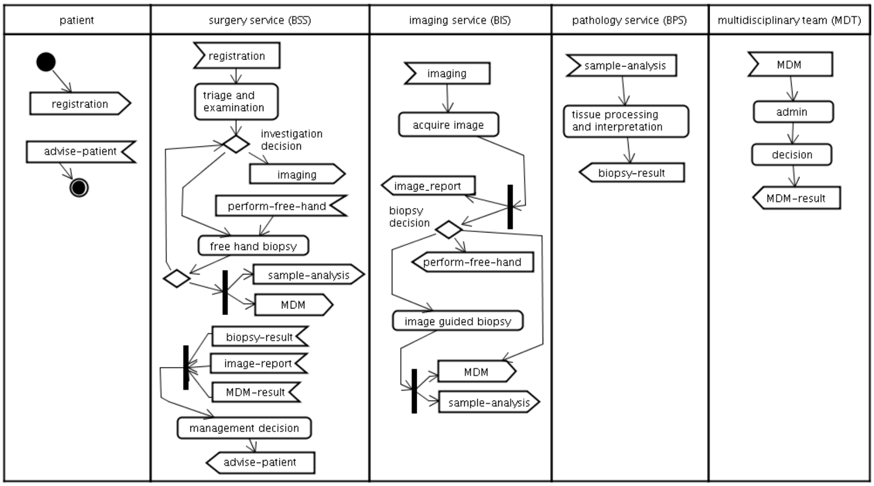

Fig. 2. Activity diagram for the triple assessment

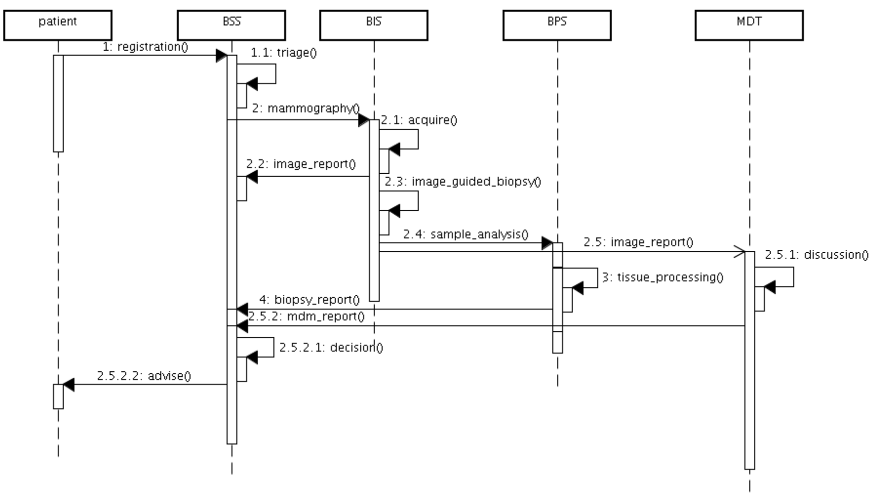

Fig. 3. Fragment of the sequence diagram of a run of the triple assessment

behaviour interface of the participants. The choreography is shared, and the participants agree to follow it. The lower level consists in the specific knowledge and services, local to participants, that are mapped to the choreography. The choreography provides a specific context and the semantics for the interaction, giving the boundaries of the task of integrating the different participants.

5 The OpenKnowledge Approach

5.1 Sharing Choreographies

A choreography-based architecture can help developers to think about distributed applications from a different perspective, one which scales better with an increasing number of interacting actors. A distributed application becomes a set of interactions between actors that assume different roles, their internal behaviour is kept separated from their external behaviour.

In most distributed approaches, the workflow engineer selects the participants: the list of participants is a basic construct both in choreography languages such as WS-CDL [23], BPEL4Chor [11] and in orchestration languages such as BPEL [2]. Some workflow architectures, such as Taverna [22], let the developer specify a list of alternative services to call if the first fails, but the binding is still at design time. Service descriptions are stored in shared repositories and the designer looks up the specific service for a task.

This approach makes portability more difficult. As we have seen, guidelines are usually written by a committee of experts with a view to being generally applicable: when the plan has to be implemented by adopting institutions, it may require heavy customisation to adapt it to their requirements (for example, the list of services to invoke is likely to be different, and therefore part of the specification has to be changed).

In our approach, the choreographies are published in a shared repository, similarly to the publication of a guideline by a committee. Participants search the appropriate choreographies when they need to perform an activity that requires the coordinated activity of other actors. The choreographies are matched against the participants' capabilities: the participants select the choreography that best fit them and advertise their intention to perform a role within a choreography. If all the roles are filled, the interaction can start. In our triple assessment example, this means that roles such as the breast imaging service or the breast pathology service are bound at run time, when the procedure needs to be performed, based on their availability.

We believe that this allows increased reuse both of choreographies and of participants' components, and allows the deployment of distributed applications with different degrees of freedom, maintaining the same architecture. Without altering the architecture presented in this Chapter, it is possible to implement a closed system where for a task there is only one possible choreography and a fixed set of other participants that are bound to it, and an open system where for a task a participant has the freedom of choosing different choreographies, and has a choice between different participants who are free to accept.

In the choreography model, issues of heterogeneity and brokering need to be addressed. Participants are likely to be different and they need to understand one another. The same services may be available from many peers, and the search and discovery process can be complex, especially if it needs to be performed at real time. The OpenKnowledge framework addresses these issues.

5.2 The OpenKnowledge Framework

The OpenKnowledge kernel [30] provides the middleware that assorted services can use to interact using a choreography-based architecture able to deal both with the semantic heterogeneity of the actors and with their discovery. It has been designed with the goals of being lightweight and compact, allowing a short development cycle for distributed applications.

$$
\begin{aligned}
Model &:= \{Clause, \ldots\} \\
Clause &:= Role :: Def \\
Role &:= a(Type, Id) \\
Def &:= Role \mid Message \mid Def \; then \; Def \mid \\
&\quad\; Def \; or \; Def \\
Message &:= M \Rightarrow Role \mid M \Rightarrow Role \leftarrow C \mid \\
&\quad\; M \Leftarrow Role \mid C \leftarrow M \Leftarrow Role \\
C &:= Constant \mid P(Term, \ldots) \mid \neg C \mid C \wedge C \mid \\
&\quad\; C \vee C \\
Type &:= Term \\
Id &:= Constant \mid Variable \\
M &:= Term \\
Term &:= Constant \mid Variable \mid P(Term, \ldots) \\
Constant &:= \text{lower case character sequence or number} \\
Variable &:= \text{upper case character sequence or number}
\end{aligned}
$$

Fig. 4. LCC syntax

The framework allows a direct translation of a choreography oriented design into an executable application. The core concept is the shared *interaction models* (IM), performed by different applications and service providers. These actors are the participants, called *peers*, of the interactions, and they play *roles* in them. In an interaction all the roles have equal weight; the behaviour of all the peers and in particular their exchange of messages are specified.

IMs are written in the Lightweight Coordination Calculus (LCC) [28,29], a compact, executable choreography language based on process calculus. Its syntax is shown in Figure 4. The IMs are published by the authors on the *distributed discovery service* (DDS) with a keyword-based description [24].

An IM in LCC is a contract that defines the expected, externally observable, behavioural interfaces of the roles that participants can take in the interaction: an IM is a set of role clauses. Participants in an interaction take their *entry-role* and follow the unfolding of the clause specified using combinations of the sequence operator

```
a(bss,BSS) ::
registration(P) ⇐ a(patient,P) then
null ← triage(TR,P) and examine(TR,ER,P) then
 ( null ← isImagingDecision(ER) then
   doImaging(P,TI) ⇒ a(bis,BIS)
      ← typeOfImaging(ER,TI) )
or
 ( ( ( null ← isFreeHandBiopsy(ER)
       then
       null ← doFreeHandBiopsy(P,FHBR) )
     or
     null ← true )
   then
   ( null ← isBiopsy(ER) then
     doSampleAnalysis(P) ⇒ a(bps,BPS) then
     doMDM(P,TR,ER,FHBR) ⇒ a(mdt,MDT) ) )
then
a(bss_wait_resports([],Reports),BSS)
then
null ← mngmt_decision(P,TR,ER,FHBR,Reports,Dec)
then
advise(Dec) ⇒ a(patient,P)

a(bss_wait_reports(RPT,NewRPT),BSS) ::
null ← all_arrived(RPT) and NewRPT = RPT
or ( biopsy_report(P,BR) ⇐ a(bps,BPS) then
     a(bss_wait_reports([BR|RPT],NewRPT) )
or ( mdm_report(P,MR) ⇐ a(mdt,MDT) then
     a(bss_wait_reports([MR|RPT],NewRPT) )
or ( image_report(P,IR) ⇐ a(bis,BIS) then
     a(bss_wait_reports([IR|RPT],NewRPT) )
```

Fig. 5. LCC clauses for the `breast surgery service(BSS)` role

(‘*then*’) or choice operator (‘*or*’) to connect messages and changes of role. Messages are either outgoing to (‘⇒’) or incoming from (‘⇐’) another participant in a given role. A participant can take, during an interaction, more roles and can recursively take the same role (for example when processing a list). Message input/output or change of role is controlled by constraints. In its definition, LCC makes no commitment to the method used to solve constraints - so different participants might operate different constraint solvers.

Figures 5 and 6 show the LCC clause for the breast surgery service (`BSS`) and the breast imaging service (`BIS`) roles in the triple assessment procedure described in the activity diagram of Figure 2. In the surgery service clause, the participant who takes the `bss` role starts waiting for the registration message from a patient. When the request message arrives, it executes the triage and the examination on the patient. Based on the result of the examination, the surgery service either asks the imaging service to acquire an image, or performs a free hand biopsy, asks the pathologist

$$
\begin{array}{l}
\texttt{a(bis,BIS)} :: \\
\texttt{doImaging(P,TI)} \Leftarrow \texttt{a(bss,BSS) then} \\
\texttt{null} \leftarrow \texttt{acquire\_image(P,TI,IMR) then} \\
\texttt{image\_report(P,IMR)} \Rightarrow \texttt{a(bss\_wait\_reports,BSS)} \\
\texttt{then} \\
\left(\begin{array}{l}
\texttt{null} \leftarrow \texttt{isBiopsyDecision(IMR) then} \\
\texttt{null} \leftarrow \texttt{doBiopsy(P,IMR,S) then} \\
\texttt{doSampleAnalysis(P)} \Rightarrow \texttt{a(bps,BPS) then} \\
\texttt{doMDM(P,TR,ER,FHBR)} \Rightarrow \texttt{a(mdt,MDT)}
\end{array}\right) \\
\texttt{or} \\
\texttt{null} \leftarrow \texttt{true}
\end{array}
$$

Fig. 6. LCC clause for the `breast imaging service (BIS)` role

to analyse the sample, and sends a request for a multi-disciplinary meeting. Then the process changes role, and waits for the reports from the contacted specialists. When all the reports have arrived, the process returns to the main role and makes the decision about the patient.

In the imaging service clause, the actor taking the role starts by waiting for a request `doImaging(P,TI)` from the surgery service. Then it acquires and analyses an image using the method specified in `TI` (either an ultrasound scan, or a mammography), and sends the report to the surgery service. If required it performs a biopsy, asks a pathologist to analyse the sample and sends a report to the multi-disciplinary team for discussion.

Most of the constraints, such as `triage(P,TR)` or `doFreeHandBiopsy (P,FHBR)`, correspond to external activities, that a doctor or a radiologist perform. They may be completed by filling in computerised forms or by receiving data from an external device such as an ultrasound scan machine. Some constraints, such as `acquire(P,TI,IMR)` may launch further interactions (in this case, a different one depending on the method required, and involving a radiologist).

LCC prescribes only the ordering of the exchanged messages and their pre and post-conditions in the form of constraints peers need to solve. However, in OpenKnowledge it is possible to annotate every element in the IM in order to enrich the description of the interaction. Annotations are mainly used to define the ontological type of the variables in messages and constraints, but time-out for messages and constraints could be expressed using the same mechanism. Figure 7 describes the syntax

$$
\begin{aligned}
annotation &:= @annotation(about, innerAnnot) \\
about &:= @role(Role)|@message(M)| \\
&\quad\; @constraint(Term)| \\
&\quad\; @variable(Variable) \\
innerAnnot &:= annotation|tree \\
tree &:= Constant|tree|Constant, tree
\end{aligned}
$$

Fig. 7. Annotations syntax

Constraint annotations
```
@annotation(@role(bss),
  @annotation(@variable(TR),
   risk_level)
)
@annotation(@role(bss),
  @annotation(@variable(P),
   patient(name,surname,date_of_birth,
    address(street,post_code)) )
)
```

Java method annotation
```
@MethodSemantic(language=''tag'',
params={
  ''patient(family_name,birthday,street,post_code)'',
  ''risk(assessed_level,confidence)''
})
public boolean doTriage(Argument P,
          Argument TR)
{...}
```

Fig. 8. Annotations for the constraint *triage(P,TR)* and for a corresponding method

used in annotations. An example of annotation for the variables in role `bss` is shown in Figure 8. This specific example shows how variables can be structured terms: the variable *P* is, in the choreography, the structure `patient(name,surname, date_of_birth, address(street,post_code)))`. The structure is a short-hand for an XML-like schema, where the functions and the parameters are nested tags in a tree. Annotations are always relative to a specific role-clause in the interaction: the scope of a variable is always limited to a clause.

Annotations are conceptually separated from the IM itself. It is possible to attach different annotations to the same IM to adapt it to different requirements and contexts: for example, the same abstract IM could be used in a different community, that specifies patients by their insurance numbers, and therefore annotates variable *P* differently.

A peer that wants to perform some task, such as providing an imaging service for breast cancer screening, searches for published IMs for the task by sending a keyword-based query to the DDS. The DDS collects the published IMs matching the description (the keywords are extended adding synonyms to improve recall) and sends the list back to the peer, that needs to choose the one to subscribe.

In open systems, IMs and peers may be designed by different entities, and therefore the constraints and the peers' knowledge bases are unlikely to correspond perfectly. The heterogeneity problem is dealt with in three phases. We have already seen the first phase: the DDS matches the interaction descriptions using a simple query expansion mechanism. Then the peers compare the constraints in the received IMs with their own capabilities [20], and finally the peers need to map the terms appearing in constraints and introduced by other peers [9]. The scope of the matching problem is limited to the specific IM in the second phase, and to the specific interaction run in the third phase.

The peer capabilities are provided by plug-in components, called OKCs (Open-Knowledge Components) that can be published by the developers on the DDS and downloaded by other peers. An OKC exposes a set of Java methods that are compared to the constraints in the IMs. The arguments of methods can be annotated similarly to variables in the IM. Arguments, like variables in constraints, can be structured terms. Figure 8 shows an annotated method with structured arguments. As annotations are short-hands for XML-like trees, values in an argument are accessed by their path, similarly to a very simplified XPath: for example, to obtain the street of a patient, the method `doTriage` will call the method `getValue(''/street'')` of the argument P.

The peer matches the annotated signatures of the constraints and of the methods, transforming them into trees and verifying their distance [15,20]. The comparison process creates *adaptors*, that bridge the constraints to the methods, as shown in Figure 9. An adaptor has a confidence level, reflecting the distance between the constraints and the matching method, that gives a measure of how well the peer can execute an interaction, and it used to select the most fitting IM. Once the peer has selected an IM, it subscribes to its role in the discovery service. Figure 10 shows a snapshot of network status when roles in an interaction are subscribed by at least one peer.

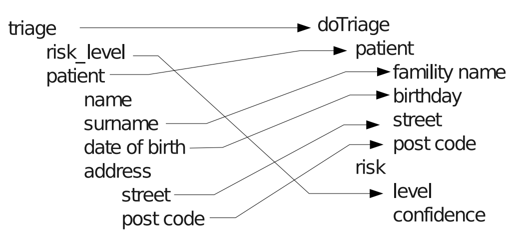

Fig. 9. Adaptor for constraint *deliver(M,P)*

When all the roles are filled, the discovery service chooses randomly a peer in the network as coordinator for the interaction, and hands over the IM together with the list of involved peers in order to execute it.

The coordinator first asks each peer to select the peers they want to interact with, forming a mutually compatible group of peers out of the replies. The selection process is subjective to the peers. All the participants receive the list of peers subscribed to all the roles, and they can check the subscriptions, selecting the preferred ones. A peer can also select none of the participants, excluding itself from a particular run of the interaction, for instance due to overload. The framework provides only an interface for the selection method, as its implementation is delegated to the application developer. However, different strategies have been tested, as we will see in the evaluation section.

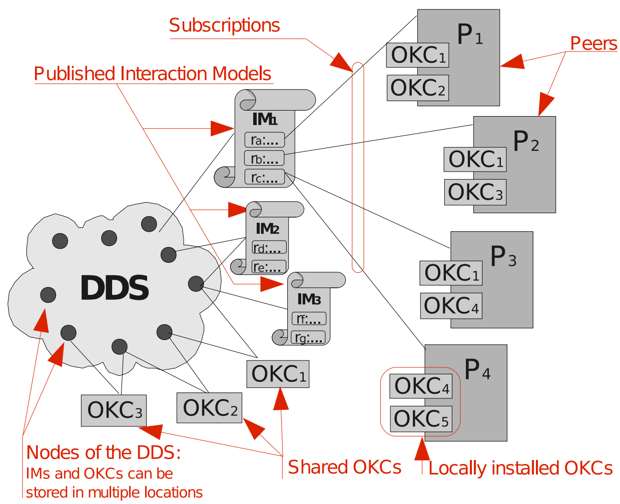

Fig. 10. OpenKnowledge architecture

While different implementations are possible, in the current version of the OpenKnowledge kernel the coordinator executes the interaction, instantiating a local proxy for each peer. The remote peers are contacted only to solve constraints in the role they have subscribed.

6 Related Work

This Section discusses all related work from the literature, spanning pure choreography languages, enhancements to widely used modelling techniques, i.e. BPMN, decentralised orchestration, data flow optimisation architectures and Grid toolkits.

6.1 Choreography Languages

The majority of workflow research has focused on designing languages for and implementing service orchestrations, from the view of a single participant. Examples can be seen in the Business Process Modelling community through BPEL [32] and life sciences community through Taverna [25]. However, there are relatively few languages targeted specifically at service choreography, the most widely known are:

• WS-CDL: The Web Services Choreography Description Language (WS-CDL) is the proposed standard for service choreography, currently at the W3C Candidate Recommendation stage. However, WS-CDL has met criticism [7,13] through the Web services community. It is not within the scope of this Chapter to provide a detailed analysis of the constructs of WS-CDL, this research has already been presented [17]. However, it is useful to point out the key criticisms with the language: WS-CDL choreographies are tightly bound to specific WSDL interfaces, WS-CDL has no multi-party support, no agreed formal foundation, no explicit graphical support and few or incomplete implementations.

• Let's Dance [36]: is a language that supports service interaction modelling both from a global and local viewpoint. In a global (or choreography) model, interactions are described from the viewpoint of an ideal observer who oversees all interactions between a set of services. Local models, on the other hand focus on the perspective of a particular service, capturing only those interactions that directly involve it. Using Let's Dance, a choreography consists of a set of interrelated service interactions which correspond to message exchanges. Communication is performed by an actor playing a role. Interaction is specified using one of three Let's Dance constructs: *precedes* – the source interaction can only occur after the target interaction has occurred; *inhibits* – denotes that after the source interaction has occurred, the target interaction can no longer occur, and finally *weak precedes* – denotes that the target interaction can only occur after the source interaction has reached a final status, e.g. completed or skipped. A complete overview of the Let's Dance language is presented in [36], including solutions to the Service Interaction Patterns [8].

• BPEL4Chor [12]: is a proposal for adding an additional layer to BPEL to shift its emphasis from an orchestration language to a complete choreography language. BPEL4Chor is a simple, collection of three artifact types: *participant behaviour descriptions* define the control flow dependencies between activities, in particular between communication activities, at a given participant. A *participant topology* describes the structural aspects of a choreography by specifying participant types, participant references and message links; this serves as the glue between the participant behaviour descriptions. Finally *participant groundings* define the technical configuration details, the choreography becomes Web service specific, concrete links to WSDL definitions and XSD types are established. BPEL4Chor is an effective proposal and importantly conforms to standards [4] by enhancing the industrially supported BPEL specification. BPEL4Chor encourages reuse by only providing a specific Web service mapping in the participant grounding. Furthermore, unknown numbers of participants can be modelled, not possible with WS-CDL.

6.2 Modelling Support

There are several proposals for extending the Business Process Modelling Notation [1]; the de-facto standard for business process modelling. Although the BPMN allows an engineer to define choreographies through a swim lane concept and a distinction between control flow and message flow, it only provides direct support for a limited set of the Service Interaction Patterns and not some of the more advanced choreography scenarios. [10] introduces a set of extensions for BPMN which facilitate an interaction modelling approach as opposed to modelling interconnected interface behaviour models. Authors claim that choreography designers can understand models more effectively, introduce less errors and build models more efficiently. Evaluation concludes that the majority of the Service Interaction Patterns can be expressed with the additional extensions. [14] discusses the deficiencies of the BPMN for choreography modelling and proposes a number of direct extensions for the BPMN which overcome these limitations.

6.3 Techniques in Data Flow Optimisation

There are a limited number of research papers which have identified the problems associated with centralised service orchestration. For completeness, this Section presents an overview of a number of architectures.

• Service Invocation Triggers [37] are also a response to the problem of centralised orchestration engines when dealing with large-scale data sets. Triggers collect the required input data before they invoke a service, forwarding the results directly to where the data is required. For this decentralised execution to take place, a workflow must be deconstructed into sequential fragments which contain neither loops nor conditionals and the data dependancies must be encoded within the triggers themselves. Before execution can begin the input workflow must be deconstructed into sequential fragments, which cannot contain loops and must be installed at a trigger.

• The *Circulate* Architecture [6] maintains the robustness and simplicity of centralised orchestration, but facilitates choreography by allowing services to exchange data directly with one another. Performance analysis [5] concludes that a substantial reduction in communication overhead results in a 2–4 fold performance benefit across all workflow patterns. An end-to-end pattern through the Montage workflow (a benchmark for the HPC community) results in an 8 fold performance benefit and demonstrates how the advantage of using the architecture increases as the complexity of a workflow grows.

• Decentralised Web Service Orchestrations Using WS-BPEL. In [38] the scalability argument made in this paper is also identified. The authors propose a methodology for transforming the orchestration logic in BPEL into a set of individual activities that coordinate themselves by passing tokens over shared, distributed tuple spaces. The model suitable for execution is called Executable Workow Networks (EWFN), a Petri nets dialect.

• Triana [39] is an open-source problem solving environment. It is designed to define, process, analyse, manage, execute and monitor workflows. Triana can distribute sections of a workflow to remote machines through a connected peer-to-peer network. *OGSA-DAI [40]* middleware supports the exposure of data resources on to Grids and facilitates data streaming between local OGSA-DAI instances. *Grid Services Flow Language (GSFL) [41]* addresses some of the issues discussed in this paper in the context of Grid services, in particular services adopt a peer-to-peer data flow model.

7 Conclusion and Future Work

In this Chapter we have argued that a distributed, choreography-based paradigm has a number of practical benefits when applied to design and implementation of complex systems, such as distributed clinical workflows. The choreography paradigm provides a clean approach to issues of portability of workflow specification (allowing a high-level guideline to be implemented at different sites which are likely to have different local procedures for carrying out tasks within the guideline), re-use of process specifications (allowing site-specific implementation detail to be re-used within different high-level process specifications), and abstraction away from the specific entities providing services, which might change both between sites and between runs of the process.

OpenKnowledge provides an operational framework for quickly setting up such systems. In OpenKnowledge, systems are composed around interaction models that coordinate the peers' behaviours by specifying the roles they can take, the exchange of messages between the roles and the constraints of the messages. Peers participate in the interaction taking one (or more) roles: in order to participate they need to compare the constraint in the roles with their available services and subscribe to the interaction on a distributed discovery service, that initiates interactions when their roles are filled.

In the current version of OpenKnowledge, constraints in the choreography are semantically matched with the capabilities of the peers. However, while feasible using the annotations, there is still no support for specifying requirements on how the constraints should be solved, or on what requirements the participants should have (for example, the peer taking the doctor role should be certified by the competent institution). Improving the specifications can help the peers both in selecting the proper interactions for their goals and in matching their capabilities with those required by the choreography.

References

1. Business Process Modeling Notation (BPMN) Specification. Technical report, Object Management Group, OMG (2006), http://www.omg.org/spec/BPMN/1.1/PDF
2. Web services business process execution language version 2.0 (April 2007)
3. Anadiotis, G., Kotoulas, S., Lausen, H., Siebes, R.: Massively scalable web service discovery. In: Proceedings of the 23rd Intl. Conference on Advanced Information Networking and Applications, AINA 2009 (2009)
4. Barker, A., van Hemert, J.: Scientific Workflow: A Survey and Research Directions. In: Wyrzykowski, R., et al. (eds.) PPAM 2007. LNCS, vol. 4967, pp. 746–753. Springer, Heidelberg (2008)
5. Barker, A., Weissman, J.B., van Hemert, J.: Eliminating the Middle Man: Peer-to-Peer Dataflow. In: HPDC 2008: Proceedings of the 17th International Symposium on High Performance Distributed Computing, pp. 55–64. ACM, New York (June 2008)
6. Barker, A., Weissman, J.B., van Hemert, J.: Orchestrating Data-Centric Workflows. In: The 8th IEEE International Symposium on Cluster Computing and the Grid (CCGrid), pp. 210–217. IEEE Computer Society, Los Alamitos (May 2008)
7. Barros, A., Dumas, M., Oaks, P.: A Critical Overview of the Web Services Choreography Description Language (WS-CDL). BPTrends Newsletter 3 (March 2005)
8. Barros, A., Dumas, M., ter Hofstede, A.: Service Interaction Patterns. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 302–318. Springer, Heidelberg (2005)

9. Besana, P., Robertson, D.: How service choreography statistics reduce the ontology mapping problem. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 44–57. Springer, Heidelberg (2007)
10. Decker, G., Barros, A.: Interaction Modeling Using BPMN. In: CBP 2007. LNCS, vol. 4928, pp. 206–217. Springer, Heidelberg (2007)
11. Decker, G., Kopp, O., Leymann, F., Weske, M.: BPEL4Chor: extending BPEL for modeling choreographies. In: Society, I.C. (ed.) Proceedings of the IEEE 2007 International Conference on Web Services (ICWS 2007), Salt Lake City, Utah, USA, pp. 296–303. IEEE Computer Society, Los Alamitos (July 2007)
12. Decker, G., Kopp, O., Leymann, F., Weske, M.: BPEL4Chor: Extending BPEL for Modeling Choreographies. In: Proceedings of the IEEE 2007 International Conference on Web Services (ICWS 2007), pp. 296–303 (2007)
13. Decker, G., Overdick, H., Zaha, J.M.: On the Suitability of WS-CDL for Choreography Modeling. In: Proceedings of Methoden, Konzepte und Technologien für die Entwicklung von dienstebasierten Informationssystemen (EMISA), LNI P-95, Hamburg, Germany, pp. 21–34 (October 2006)
14. Decker, G., Puhlmann, F.: Extending BPMN for Modeling Complex Choreographies. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 24–40. Springer, Heidelberg (2007)
15. F., G., M., Y., F., M.: Structure preserving semantic matching. In: Proceedings of the ISWC+ASWC International workshop on Ontology Matching (OM), Busan, KR (2007)
16. Grimshaw, J.M., Russel, I.T.: Effect of clinical guidelines on medical practice: a systematic review of rigorous evaluations. Lancet 342, 1317–1322 (1993)
17. Fredlund, L.: Implementing WS-CDL. In: Proceedings of the second Spanish workshop on Web Technologies (JSWEB 2006), Universidade de Santiago de Compostela (November 2006)
18. Fox, J., Patkar, V., Thompson, R.: Decision support for health care: the proforma evidence base. Journal of Informatics in Primary Care 14(1), 49–54 (2006)
19. Garg, A., Adhikari, N., McDonald, H., Rosas-Arellano, P., Devereaux, P., Beyene, J., Sam, J., Haynes, R.: Effects of computerised clinical decision support systems on practictioner performance and patient outcome: a systematic review. JAMA 293, 1223–1238 (2005)
20. Giunchiglia, F., McNeill, F., Yatskevich, M., Pane, J., Besana, P., Shvaiko, P.: Approximate structure preserving semantic matching. In: On the Move to Meaningful Internet Systems: OTM 2008, pp. 1217–1234 (2008)
21. Grimshaw, J., Russel, I.: Effect of clinical guidelines on medical practice: A systematic review of rigorous evaluations. Lancet 342, 1317–1322 (1993)
22. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M.R., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Res. 34(Web Server issue), W729–W732 (2006)
23. Kavantzas, N., Burdett, D., Ritzinge, G., Fletcher, T., Lafon, Y., Barreto, C.: Web services choreography description language version 1.0 (November 2005), `http://www.w3.org/TR/2005/CR-ws-cdl-10-20051109/`
24. Kotoulas, S., Siebes, R.: Deliverable 2.2: Adaptive routing in structured peer-to-peer overlays. Technical report, OpenKnowledge
25. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20, 3045–3054 (2004)

26. Patkar, V., Hurt, C., Steele, R., Purushotham, A., Williams, M., Thomson, R., Fox, J.: Evidence-based guidelines and decision support services: a discussion and evaluation in triple assessment of suspected breast cancer. British Journal of Cancer 95, 1490–1496 (2006)
27. Perreau de Pinninck, A., Sierra, C., Walton, C., De la Cruz, D., Robertson, D., Gerloff, D., Jaen, E., Li, Q., Sharman, J., Abian, J., Schorlemmer, M., Besana, P., Leung, S., Quan, X.: D6.4: Summative report on bioinformatics case studies. Technical report, University of Edinburgh (2008)
28. Robertson, D.: Multi-agent coordination as distributed logic programming. In: International Conference on Logic Programming, Sant-Malo, France (2004)
29. Robertson, D., Walton, C., Barker, A., Besana, P., Chen-Burger, Y., Hassan, F., Lambert, D., Li, G., McGinnis, J., Osman, N., Bundy, A., McNeill, F., van Harmelen, F., Sierra, C., Giunchiglia, F.: Models of interaction as a grounding for peer to peer knowledge sharing. In: Advances in Web Semantics I: Ontologies, Web Services and Applied Semantic Web, vol. 4891, pp. 81–129 (2009)
30. Siebes, R., Dupplaw, D., Kotoulas, S., de Pinninck, A.P., van Harmelen, F., Robertson, D.: The openknowledge system: an interaction-centered approach to knowledge sharing. In: Proceedings of the 15th Intl. Conference on Cooperative Information Systems, CoopIS (2007)
31. Sutton, R., Fox, J.: The syntax and semantics of the proforma guideline modeling language. Journal of the American Medical Informatics Association 10(5), 433–443 (2003)
32. The OASIS Committee: Web Services Business Process Execution Language (WS-BPEL) Version 2.0 (2007)
33. Trecarichi, G., Rizzi, V., Vaccari, L., Marchese, M., Besana, P.: Openknowledge at work: exploring centralized and decentralized information gathering in emergency contexts. In: ISCRAM 2009 (2009)
34. Vaccari, L., Shvaiko, P., Marchese, M.: An emergent semantics approach to semantic integration of geo-services and geo-metadata in spatial data infrastructures. International Journal of Spatial Data Infrastructured Research 4, 24–51 (2009)
35. var der Aalst, W., Aldred, L., Dumas, M., ter Hofstede, A.H.M.: Design and implementation of the yawl system. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 142–159. Springer, Heidelberg (2004)
36. Zaha, J.M., Barros, A., Dumas, M., ter Hofstede, A.: Let's Dance: A Language for Service Behavior Modelling. In: Meersman, R., Tari, Z. (eds.) OTM 2006. LNCS, vol. 4275, pp. 145–162. Springer, Heidelberg (2006)
37. Binder, W., Constantinescu, I., Faltings, B.: Decentralized Ochestration of Composite Web Services. In: Proccedings of the International Conference on Web Services, ICWS 2006, pp. 869–876. IEEE Computer Society, Los Alamitos (2006)
38. Martin, D., Wutke, D., Leymann, F.: A Novel Approach to Decentralized Workflow Enactment. In: EDOC 2008. 12th International IEEE Conference on Enterprise Distributed Object Computing, pp. 127–136 (2008)
39. Taylor, I., Shields, M., Wang, I., Philp, R.: Distributed P2P Computing within Triana: A Galaxy Visualization Test Case. In: 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), pp. 16–27. IEEE Computer Society, Los Alamitos (2003)
40. Karasavvas, K., Antonioletti, M., Atkinson, M., Hong, N.C., Sugden, T., Hume, A., Jackson, M., Krause, A., Palansuriya, C.: Introduction to OGSA-DAI Services. In: Herrero, P., S. Pérez, M., Robles, V. (eds.) SAG 2004. LNCS, vol. 3458, pp. 1–12. Springer, Heidelberg (2005)
41. Krishnan, S., Wagstrom, P., Laszewski, G.v.: GSFL: A Workflow Framework for Grid Services. Technical report, Argonne National Argonne National Laboratory (2002)