

# Knowledge Representation Environments: an Investigation of the CASSMs between Creators, Composers and Consumers

Ann Blandford<sup>1</sup>, Thomas R. G. Green<sup>2</sup>, Iain Connell<sup>1</sup> and Tony Rose<sup>3</sup>

<sup>1</sup> UCL Interaction Centre, University College London, Remax House, 31-32 Alfred Place  
London WC1E 7DP, U.K.

A.Blandford@ucl.ac.uk

<http://www.ucl.ac.uk/ucl-icc/annb/>

<sup>2</sup> University of Leeds, U.K.

<sup>3</sup> System Concepts Ltd., U.K.

**Abstract.** Many systems form ‘chains’ whereby developers use one system (or ‘tool’) to create another system, for use by other people. For example, a web development tool is created by one development team then used by others to compose web pages for use by yet other people. Little work within Human–Computer Interaction (HCI) has considered how usability considerations propagate through such chains. In this paper, we discuss three-link chains involving people that we term Creators (commonly referred to as designers), Composers (users of the tool who compose artefacts for other users) and Consumers (end users of artefacts). We focus on usability considerations and how Creators can develop systems that are both usable themselves and also support Composers in producing further systems that Consumers can work with easily. We show how CASSM, an analytic evaluation method that focuses attention on conceptual structures for interactive systems, supports reasoning about the propagation of concepts through Creator–Composer–Consumer chains. We use as our example a knowledge representation system called Tallis, which includes specific implementations of these different perspectives. Tallis is promoting a development culture within which individuals are empowered to take on different roles in order to strengthen the ‘chain of comprehension’ between different user types.

**Keywords:** usability evaluation methods; CASSM; design chains

## 1 Introduction

It is widely recognised that there are many stakeholder groups in any design project, typically including managers, purchasers, end users and developers. Approaches such as Soft Systems Methodology [5] encourage an explicit consideration of these different stakeholder groups in design. However, when it comes to considering usability, the focus narrows immediately to the ‘end users’ of the system under consideration. For example, most classic evaluation techniques, such as Heuristic Evaluation [13] and Cognitive Walkthrough [16] focus on what the user will

experience in terms of their tasks and the feedback received from the system. Norman [15] discusses the relationship between designer and user in terms of the ‘designer’s conceptual model’ – an understanding of the system that has to be communicated from designer to end user via the interface. In all these cases, the focus remains on a single ‘system’ that is being designed. No work that we are aware of extends this perspective. This is perhaps not surprising, since established usability-oriented analysis techniques focus attention on user tasks and the procedures users need to follow to complete those tasks. The users of different interfaces experience interactions with different properties that are not readily related to each other. In this paper, we explicitly consider different systems within a development chain, focusing in particular on three groups of stakeholders that we term Creators, Composer and Consumers (C<sup>3</sup>) – namely the Creators of tools that can be used by Composers to construct products for Consumers to use.

As an early example, consider the design of web pages using a web composition tool such as Dreamweaver<sup>®</sup>. It is recognised good practice for all pictures on web pages to be supplemented by “ALT” text that describes the content of the picture to improve ease of use for users with limited vision. If the web composition tool makes it easy for Composers to include ALT information, and makes it obvious at the time of including a picture that ALT text should be added, then the resulting web page will be more usable. The Creator of the web development tool can improve the likely usability of web pages produced by a Composer if the Creator is aware of the potential needs of the Consumer.

We argue that a declarative approach to evaluation can yield a more insightful evaluation of such chains than a procedural one, for reasons presented below. The declarative approach we have adopted is CASSM [4], a technique for usability evaluation that is based on identifying the concepts with which users are working and those implemented within a system. This approach has helped to draw out relationships between different systems within a ‘chain’ of products that have (typically) different users and different interfaces. The approach is exemplified with a system, Tallis, for representing clinical guidelines that makes explicit the fact that it has different classes of users who experience different interfaces.

## **1.1 Creators, Composers and Consumers**

We do not consider ourselves to have invented C<sup>3</sup> chains; indeed, they are a widespread phenomenon. Nevertheless, we are not aware of prior work that has discussed such chains within the HCI literature, or considered usability in terms of chains. Therefore, we start by briefly discussing some examples of C<sup>3</sup> chains – namely website creation tools, programming development environments and online library building applications.

Website creation tools such as Dreamweaver<sup>®</sup> allow Composers to create, edit and manipulate html and other mark-up language code prior to uploading finished website code to a server. The role of Creators is not only to make the programming and editing environment easy to use but also to facilitate the creation of usable, acceptable web sites (as illustrated with the ALT text example above). The role of Composers is to create and test web pages or sites that are easy and pleasant for the end user to

work with. The role of Consumers is to browse, search or otherwise work with the resulting web sites. Thus, the Creators have to understand not only what Composers will experience, and consider the usability of the composition environment, but also make it easy for Composers to deliver web pages that are well laid out and easy for Consumers to interact with.

The same roles are to be seen in the design and use of program development environments such as NetBeans, an interactive Java environment [13]. The Creators of NetBeans and similar environments provide a tool that will meet the needs of Composers, i.e. Java programmers. Composers write Java programs that should be usable by the Consumers — the people who work with those programs to get a job done. In some cases the roles may become blurred: the same person may create the environment, use it to write programs, and then make use of those programs; nevertheless, there are separate roles depending on which system is the current focus of use.

Chains may stretch further in both directions: a Java programming environment may be written in another programming language, say C++, for which a compiler may be written in some other language—stretching back through assembly code to the instruction set recognised by the hardware. In the other direction, Java programs created in NetBeans, etc, may be used as tools by people who are building other tools. Chains may also branch; for example, web applications are viewed in browsers, and there are often interactions between the application and the browser itself so that the design of both influences the users' experience. This is a factor in the design of Tallis as discussed below, but we do not consider this branching further in this paper.

An example of a tool that extends the development chain is a digital library system, where developers work with software development environments to create a further layer of tools, such as Greenstone [17], with which librarians can create collections of documents to be made available to end users. In a study evaluating the Greenstone digital library software [2], one of the developers commented as follows:

“[There is a] difficulty with the way Greenstone is perceived by different parties. [The developers] see Greenstone very much as a toolset which other folks should 'finish off' to make a good collection. Their conception is that it would be very hard to take Greenstone to a level where a librarian could make a few choices on GUI [Graphical User Interface] and have a reasonable (not to say actively excellent) interface for the library.”

In other words: in the view of the respondent, the Creators of the Greenstone toolset were not recognising their potential role in making it easy for Composers (who typically have little HCI expertise) to construct usable digital libraries for Consumers.

The possibility that a development environment such as NetBeans might be used to construct a digital library tool set like Greenstone, which would in turn be used to develop digital libraries, illustrates the idea that the overall chain might involve more than three groups of designers/users. Here, we only consider C<sup>3</sup> chains. Within a longer chain, the decision as to which people fill the roles of Creator, Composer, and Consumer would depend on where the focus of interest is. In the case of NetBeans, it would be on the development environment and resulting systems, whereas for Greenstone it would be on the development tools and resulting library collections.

Table 1 tabulates the distinction between Creators, Composers and Consumers for these and other systems. In all these cases, there will typically be a development team who create the tool; they may or may not have direct access to their immediate users,

the Composers of products. The end users (Consumers) of the product typically have a role where interaction is relatively constrained, with limited scope for changing structures within the product.

**Table 1.** Distinction between Creators, Composers and Consumers for different types of interactive system.

Creator of tool	Composer of product	Consumer of product
User Interface Development Environment	Develop interfaces	Use interfaces
Online library tool set (e.g. Greenstone)	Create and manage library collections	Retrieve and display search items
Drawing tool	Create and edit drawings	View and interpret drawings
Website creation tool (e.g. Dreamweaver)	Create and manipulate html and other code, run web pages in a browser	Run website in a browser
Programming development system (e.g. NetBeans)	Create and manipulate code, test programs, run programs	Run programs
Music composition system	Create and edit musical representations	Read, interpret and play music
Word processing system	Create and edit text	Read and interpret text
Game engine	Create new game software	Play game

## 1.2 CASSM and Misfit Analysis

With this understanding of  $C^3$  systems, we turn to consider evaluation of these different systems. Approaches to the evaluation of any interactive system, whether analytic or empirical, based on prototype or working artefact, require from evaluators an insight into the assumptions and expectations held by the intended users of that system. CASSM (Concept based Analysis of Surface and Structural Misfits) is an analytic method which aids the identification of designer-user misfits. Prior to this study, it had only been used in the traditional way, of considering a single interactive system and its users. This study extends the use of CASSM to consider  $C^3$  chains.

In contrast to most evaluation approaches, CASSM does not focus on tasks, but on entities and attributes, and the differences between the system and user models of how entities and their attributes are represented and manipulated at the interface. Previously, we have described how CASSM can identify misfits in systems as diverse as drawing tools and online music libraries [7] and ambulance dispatch [3]. In this paper, we extend the application of CASSM to Tallis [8], a knowledge representation system that exhibits an unusual degree of overlap between the  $C^3$  roles. The CASSM analysis of Tallis allows us to distinguish between the useful and less useful manifestations of this overlap.

In a CASSM analysis, we make an explicit distinction between the representation embodied within an interactive system and that understood by the users of that system. Earlier papers [1,6] show how we characterise this distinction in terms of a taxonomy of User, Interface and System properties, where the various concepts

(entities and attributes) which result from the CASSM analysis are depicted as Present or Absent from the System models, and Present, Absent or Difficult to apprehend for the User or via the Interface. (See [4] for tutorial and worked examples).

In its emphasis on objects rather than tasks, CASSM is distinct from other analytic approaches which aim to illuminate the differences between system and user models. Connell *et al* [6] have contrasted CASSM with Cognitive Walkthrough, whose focus on goal support at each stage of a task has some similarities with Norman's [15] theory of action (which depicts system-user misfits in terms of the gulfs of execution and evaluation). CASSM can be viewed as focusing more on the conceptual gulfs that Norman [15] discusses between the designer and the user.

## 2. Tallis Composer and Enactor

As noted above, knowledge representation tools also exemplify the C<sup>3</sup> chain. Composers create, manipulate and edit a rule-based set of choices and actions, presented to Consumers via an interface. Tallis is a knowledge representation tool that is being developed with a view to producing and disseminating guidelines for clinical practice. It is typically used for modelling clinical diagnosis and treatment processes in the domain of Oncology (the branch of medicine that deals with cancer).

Tallis comprises three interrelated systems: Composer, Tester and Engine. Tester supports debugging, and is not considered in this study. Tallis Composer is a Graphical User interface (GUI) environment which supports the composition of guidelines to aid clinicians in diagnosis and treatment. Guidelines, the output from the Composer, are held in PROforma code [9]. Tallis Engine is the environment in which guidelines are run (or enacted). Enactment takes place in a web browser via a Java virtual machine. In this section, we describe Tallis using an illustrative (non-clinical) guideline for use of the London Underground ticket vending machines. Later sections present the results of a CASSM analysis of Tallis.

This ticket vending machines domain was chosen for two reasons. First, in order to gain experience of using Tallis, it was easier to create and test a guideline in a familiar domain. Second, for the purposes of eliciting Consumer feedback on use of a Tallis guideline, it was easier to recruit a user group who were familiar with the ticketing domain than it would have been to recruit oncology specialists.

### 2.1 Tallis and PROforma

Tallis is a Java implementation of a knowledge representation language called PROforma, which is designed to support the publication of clinical expertise [11]. Support takes the form of an expert system which assists patient care through active decision support and workflow management. Fox *et al* [10] describe PROforma as an "intelligent agent" language and technology, where agent specification is done by composing tasks into collections of prepared plans. Plans can be enacted sequentially, in parallel, or in response to events.

The PROforma decision and plan model offers four classes of task, namely Plans, Decisions, Actions and Enquiries. The root class of this structure is the Keystone, an empty ‘placeholder’ task. Decisions, Actions and Enquiries may be combined to make up Plans, which themselves consist of other tasks, including other Plans. A combination of tasks so formed represents a PROforma guideline, encapsulating one piece of clinical expertise, which may be published on a world wide web repository such as the Open Clinical Knowledge Publishing Collaboratory [12].

Figure 1 shows an extract from the Tallis Composer representation for a sample guideline to support use of London Underground ticket vending machines (TVMs).

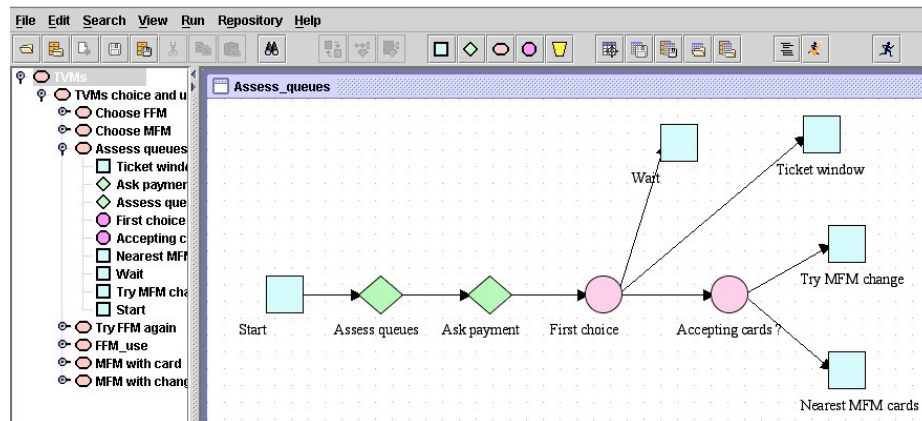


Figure 1. Extract from the Tallis Composer tool.






The left-hand panel of Figure 1 shows part of the TVMs guideline task hierarchy, and the large panel the structure of the task named Assess\_queues. The middle part of the toolbar above the panel offers the five PROforma tasks (Action , Enquiry , Plan , Decision  and Keystone ), any of which can be inserted into Assess\_queues (by drag-and-drop from the toolbar to the task window). Other panels (not shown) allow configuration of the attributes of each task component.

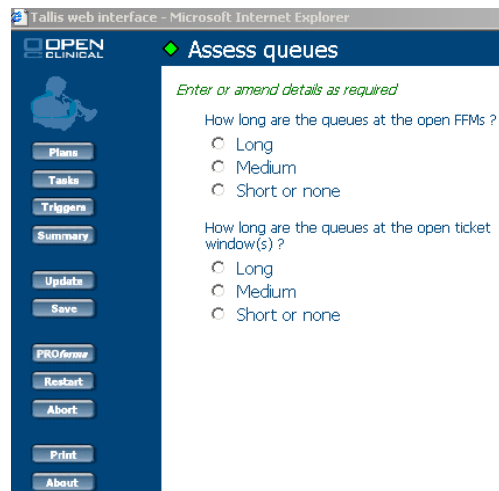
Figure 2 shows the initial result of enacting the above guideline in a web browser using Tallis Engine. The left-hand panel allows the guideline user to inspect certain components of the guideline, including the PROforma itself, and to summarise the enactment trail thus far. The guideline may also be restarted or aborted.

## 2.2 Tallis Users

In the Tallis context, *Creators* produce and design the Tallis Composer interface and also the default Tallis Engine interface (Figures 1 and 2; sophisticated Tallis users can tailor the Engine interface to suit the needs of their application). *Creators* also prescribe how the PROforma code which results from a Composer session is to be enacted. *Composers* make use of Tallis Composer to produce guidelines (or self-contained guideline fragments) which are encoded in PROforma and run via the

Engine. Composers may also publish guidelines in a Repository. *Consumers* download published guidelines and run them in a web browser using the Engine.

Knowledge representation systems such as Tallis are interesting examples of C<sup>3</sup> systems because the assumptions made by the guideline Composer about Consumer expectations and knowledge are critical to guideline usage, and it is the task of the Creator to make it easy for the Composer to easily generate usable guidelines that match the understanding of Consumers. As noted above, the Engine interface is tailorable, so the challenge might be more appropriately stated as that of producing a good general default that can be readily tailored to particular user groups.



**Figure 2.** Initial result of enacting a PROforma guideline (created using Tallis Composer) in a web browser (using Tallis Engine). The task being run is Access\_queues.

### 3. CASSM Analysis of Tallis Composer and Engine

This Section describes the result of applying the CASSM approach to Tallis Composer and Engine, and setting out the results using a dedicated tool named Cassata (available from [4]).

An important part of a CASSM analysis is the elicitation of user data. In the case of Tallis, this took complementary forms as described below. In practice, the current culture of working with Tallis meant that some participants spoke from more than one perspective; thus, most of the clinical interviewees feature below in multiple sections.

#### 3.1 Data collection

One source of data was a detailed diary, kept by the lead researcher, of insights into the experiences of learning Tallis over a period of several weeks. As discussed above,

the guideline that was developed represented knowledge about underground ticket purchase. During this time, the researcher worked closely with Tallis Creators to improve their awareness of novice user difficulties and to improve his understanding of the system design.

One of the Creators (i.e. a core member of the Tallis development team) was interviewed about his perceptions of the system. Another of the Creators was recorded while Composing the first part of a guideline for the ticket vending machines. The video protocol so obtained represents an expert view of guideline composition and Tallis Composer use, as well as giving insights into the design philosophy for Tallis. The comparison between his version of the TVMs guideline and the larger but less efficient version initially produced by the researcher was used to probe the differences between expert and novice Composers. Following this comparison, the TVMs guideline intended for Consumer use was re-composed.

Three further Tallis users were interviewed. One was an Oncology clinician who worked closely with the Tallis developers and used Tallis to create and upload sample guidelines to the CRUK repository; he was able to present the views of Creator, as well as Composer and Consumer. The second was a professor of medical informatics who had also made use of Tallis in teaching. The third was a lecturer in Health Informatics who based some of his teaching and student course work around Tallis. All three interviewees were asked about their views of using Tallis Composer and Engine to produce guidelines; with two of them, it was possible to run through sample repository guidelines. In one case the participant demonstrated how he had used Tallis to compose guidelines. In the other, the participant acted as Consumer while running the TVMs guideline, and then inspected the guideline components as Composer. The second and third participants were asked about the wider context of decision support systems, and specifically how Tallis compares with similar systems.

To obtain views of Consumers that were independent of the Composer perspective, five postgraduate HCI students used the ticket vending machine guideline to complete sample ticket-buying scenarios. They were asked about their perceptions of Tallis Engine. Interviews were audio recorded and relevant issues extracted.

### **3.2 Analysis and Results**

We present the results according to role; as outlined above, several of the study participants discussed Tallis from multiple perspectives, which we have separated out here. Because several of the interviewees had a clinical or medical informatics background, they were able to talk about their views as Consumers of Tallis enacted guidelines as well as their views as Creators or Composers; therefore, we consider two separate groups of Consumers: those of enacted clinical guidelines and those of the enacted ticket machine guideline.

We have constructed CASSM descriptions of the Tallis Composer and Tallis Engine to highlight user–system misfits. These have been constructed by working through interview transcripts and system descriptions to identify the core user and system concepts. On the user side, contextual information from transcripts has been used to determine whether those concepts are present in the user’s conceptual model of the system, whether they pose user difficulties or whether they are absent from the



user's conceptualisation. The user's conceptualisation will typically include both system (device) concepts and ones pertaining specifically to the domain in which they are working. On the interface and system side, system descriptions have been used to determine whether concepts are represented at the interface and in the underlying system model. For every concept, where possible, further data has been used to determine how easily actions can be performed to change the state of the concept (e.g. creating new entities or changing attributes). Where this data has not been available, we have entered 'not sure' in the CASSM table. More details of conducting a CASSM analysis are available from [4].

### 3.2.1 Creators

One of the interviewees from Cancer Research UK described the system as follows:

"what we are looking [at] is how to provide decision support, which will be a core of this project, over the treatment of the patient, from diagnosis until follow-up treatment and everything, so basically this is ... there are a lot of other things apart from decision support, like, urm, automatic enactment of other tasks, and lots of other things, but the core part of it is decision support for clinicians, and it will also record all data, data entry." [taken from transcript of interview]

Another compared Tallis to a flow-chart representation of clinical care pathways: the flow chart representation "has some of the same high-level goals and 'spin', and that is an approach that is very common. It's an importantly different approach, because the guidelines are not enactable. They cannot be created by clinicians and then enacted by others. There is no active decision support." [taken from handwritten notes of interview]

Thus, from a Creator perspective, Tallis is a system that supports the development and use of clinical guidelines, with important features such as active support for decision making and integration with other clinical tasks within the overall patient care pathway. One important feature, highlighted in the first of the above extracts, is that Tallis provides the facility to generate an audit trail of clinical decisions in case of any queries about the clinical decision making for a particular patient. Although the developers think of the system used within a clinical context, it is also possible to implement guidelines for other decision making tasks – such as the ticket machine example used within this study.

### 3.2.2 Composers

The following extracts from interviews highlight Composer perspectives on Tallis Composer. These perspectives have formed the basis for the CASSM description presented below. Key ideas built into the CASSM model are highlighted in yellow (or greyscale) within the transcript.

"there are multiple plans and tasks, and each plan involves another task" [1<sup>st</sup> interviewee]

"[the guideline] will support investigations ... actually this is not the latest version [of Tallis], what have added is clinical evidence." [1<sup>st</sup> interviewee]

“as an editing tool it’s very difficult to keep track of because you don’t have **global view**.” [2<sup>nd</sup> interviewee]

The second interviewee also discussed the challenge of teaching students to work with Tallis. In particular, he highlighted the idea that there are some standard ‘patterns’ of structure within a knowledge representation (typical patterns of components that represent common ways of reasoning) that can be reused when constructing large guidelines, but that students have to construct them from first principles every time:

“Students are asked to consider how they might put a **pathway** through a **set of Tallis components [Plans, Actions, Enquiries, Decisions]**. Getting more than simple ‘asking for information and using that in next decision’ combinations is difficult - we use a **pattern for a Plan that is a query and choice** depending on the answer to that query” [2<sup>nd</sup> interviewee]

The third interviewee talked about what Tallis is not as well what it is, but then repeated many of the concepts enumerated by the first interviewee:

“We don’t get support in Tallis for knowledge representation - Tallis doesn’t have (modelling) tools with which we can build a model (of e.g. a patient) from which statements can be taken. Tallis doesn’t allow you to represent the underlying model of (e.g. a patient). Tallis is not object or entity oriented (but is process or ‘task’ oriented) - you [the guideline creator] have to map decision criteria onto the ‘**objects**’ provided by Tallis (which are **plans, enquiries, decisions, actions** etc.).” [handwritten notes of 3<sup>rd</sup> interview]

To construct the full CASSM description, we can also take information from a simple system description (extracted from [9]), as follows:

“*PROforma* is a formal knowledge representation language capable of capturing the **structure and content of a clinical guideline** in a form that can be interpreted by a computer. The language forms the basis of a method and a technology for developing and publishing executable clinical guidelines. Applications built using *PROforma* software are designed to support the management of medical procedures and clinical decision making at the point of care.

In *PROforma*, a guideline application is modelled as a set of **tasks** and **data items**. The notion of a task is central - the *PROforma* task model [...] divides from the keystone (**generic task**) into four types: **plans, decisions, actions and enquiries**.

**Plans** are the basic building blocks of a guideline and **may contain any number of tasks** of any type, including other plans. **Decisions** are taken at points where **options** are presented, e.g. whether to treat a patient or carry out further investigations. **Actions** are typically clinical procedures (such as the administration of an injection) which need to be carried out. **Enquiries** are typically requests for further information or data, required before the guideline can proceed.

[...] networks of tasks can be composed that represent plans or procedures carried out over time. In the editor, logical and temporal relationships between tasks are captured naturally by linking them as required with arrows. Any procedural and medical knowledge required by the guideline as a whole or by an individual task is entered using **templates** attached to each **task**.”

These extracts do not define a full model, but are sufficient for an illustrative, sketchy CASSM model, as shown in Table 2.

**Table 2:** Entities and attributes for Tallis Composer as extracted from user data of Composers.

	Concept	User	Interface	System	Set / create	Change / delete	Notes
E	guideline	present	difficult	present	easy	easy	difficult to get an overview of the guideline
A	evidence	present	present	present	easy	easy	easy for composer, harder for engine
A	investigation	present	present	present	easy	easy	
E	task	difficult	present	present	easy	hard	Also called 'components' and 'objects'.
E	a decision pathway	present	difficult	notSure	notSure	notSure	
E	data item	present	present	present	easy	easy	
E	pattern	present	absent	absent	cant	cant	
E	plan	notSure	present	present	easy	notSure	
A	attributes	notSure	present	present	easy	notSure	
E	action	notSure	present	present	easy	notSure	
A	attributes	notSure	present	present	easy	notSure	
E	enquiry	notSure	present	present	easy	notSure	
A	attributes	notSure	present	present	easy	notSure	
E	decision	notSure	present	present	easy	notSure	
A	attributes	notSure	present	present	easy	notSure	Includes options

This CASSM description includes notes of superficial difficulties as highlighted in the interviews: that it is difficult to get an overview of a guideline at the interface, that components (and their linkages) are hard to change once created, and that the idea of a ‘pattern’ of structure is important to some Composers, but is absent from the Tallis Composer environment. This sketchy description does not account for difficulties users might experience in constructing clinical guidelines using the PROforma language – that would require a more thorough analysis than is appropriate for the present purpose.

### 3.2.3 Consumers: clinicians

As noted above, most of the clinical interviewees discussed their experiences of Tallis Engine (i.e. the Consumer interface). Their descriptions of Engine included the following from the first interviewee:

“this is - from a **patient's history**, [...] of breast cancer, and this is **examination of imaging**, of mammogram or ultrasound” [1<sup>st</sup> interviewee]

“this is the first screen which are some information about the **demographics about the patient**. There is a, some more information, and whether the patient has got a previous medical past, if you say yes, then [another part of the dialogue becomes ungreyed out], otherwise it is greyed out; here we can see that the patient is not **pregnant** and the patient has got some **family history** [...] patient has got a **lump which is 30 mm and which is not fixed** “ [1<sup>st</sup> interviewee]

“**Interventions** [in enacted guidelines] don't mean anything to clinicians - change to '**candidates**', but names of decisions should be captions, not technical names. “ [1<sup>st</sup> interviewee]

The same interviewee commented on the experience of working with Engine:

“this process [...] forces the clinician to do a particular **sequence of the task**, which in actual practice is not the case always. [...] But otherwise, for different clinicians, if you take a novice candidate or a clinician who is very junior, this probably is better because it guides the clinician [in] the normal steps. But for a senior clinician, say for a consultant, it’s sometimes irritating, like, he don’t want to go all the stages he already know, so he might go to a particular task” [transcript of 1<sup>st</sup> interviewee]

“sometimes it might inhibit a clinician - the other thing is we cannot go back, like if I enter some details here, and the **patient** came up with some other details at a later stage, [or] if I forgot to enter the **details** [earlier], I can’t go back” [transcript of 1<sup>st</sup> interviewee]

The second interviewee commented explicitly about the relationship between the Composer and Engine environments; the following refers to the Engine window:

“Top level presentation is fine - the next level down needs to be ... if the things aren’t **boolean statements**, and they are just **pieces of evidence** for and against then it’s not too bad, [but] if they’re things like this, which is a **long expression** [looking at the Interventions page, after the first pair of enquiry windows] that’s not something I’d want my users - my end users - to see. I’m perfectly happy for my knowledge engineer to see that, as part of the debugging process ... but it doesn’t display boolean combinations well at this point.” [2<sup>nd</sup> interviewee]

**Table 3:** Entities and attributes for Tallis Engine as extracted from user data of clinical users.

	Concept	User	Interface	System	Set / Create	Change/ Delete	Notes
E	guideline	present	difficult	present	fixed	fixed	difficult to get an overview of the whole guideline
A	clinical evidence	present	present	present	easy	hard	
A	investigation	present	present	present	easy	hard	
A	intervention	difficult	present	present	easy	hard	"should be called 'candidates'"
E	patient	present	notSure	notSure	easy	easy	
A	"model"	present	absent	absent	cant	cant	
A	details	present	present	notSure	easy	hard	
A	history	present	present	present	easy	hard	
A	demographics	present	present	present	easy	hard	
A	symptoms	present	present	present	easy	hard	
E	treatment	present	present	present	fixed	fixed	
E	care pathway	present	notSure	notSure	notSure	notSure	
E	plan	difficult	present	present	fixed	fixed	
E	task	difficult	present	present	fixed	fixed	
E	trigger	difficult	present	present	fixed	fixed	
E	PROforma	difficult	present	present	fixed	fixed	
E	evidence	present	present	present	easy	notSure	
A	representation	difficult	present	present	fixed	fixed	
E	decision process	present	present	present	notSure	notSure	
E	decision outcome	present	present	present	notSure	indirect	
E	decision /data record	notSure	difficult	present	indirect	cant	

The third interviewee compared Tallis to flowchart descriptions of clinical guidelines:

“Flow-chart representations [of guidelines] might be better than a Tallis representation (you just have to use your eyes to follow it). However, representations involving timelines (e.g. care pathways) might need the additional complexity of systems such as Tallis.” [notes from 3<sup>rd</sup> interview]

These extracts, together with reference to the Engine environment (as illustrated in Figure 2), have been used to construct the CASSM description shown in Table 3. This is not instantiated to a particular clinical problem (e.g. the diagnosis and treatment of breast cancer), but is a general model of clinical guideline use.

This shows more substantial likely user difficulties than the Composer environment; users are expected to work with concepts (such as ‘intervention’, ‘plan’ or ‘PROforma’) that are unfamiliar, and of minimal obvious relevance to them in their (clinical) decision making. In addition, while much information is easy to enter, it is difficult to change later, due to the linear model of decision making implemented within Tallis.

### 3.2.4 Consumers: TVMs

Many of the same issues emerge in the findings from the study of ticket machine decision making. The data for the ticket machine Consumers is taken from the implementation and user comments on the Engine guideline produced as part of this study. Extracts from user comments are as follows. In all these cases the extracts are taken from questionnaires completed after the interaction or from the analyst’s notes, and numbers at the beginning indicate which user made the comment. The first set of comments refer, as with the clinical users, to Tallis concepts that are independent of the domain of ticket purchasing:

[1] “Don’t need the ‘Intervention’ screen - it gives information that I already know.”

[5] “Interventions screens look like programming language - had to understand boolean logic to use it - seems like decision-making screen”

[1] “Don’t feel in control - have to follow path, can’t make choices that are not offered.”

[5] “Summary [at end] are titles of tracks, not what I did. Does not remind you of overall goal, nor the tracks you have done. Summary is textual way of showing the process, not the overall goal.”

[1] “Can’t use the summary [trail of previously used Tallis entities] to go back to previous stages [in order to do alternative forward routes without having to restart]”

[5] “Not sure if Print will reproduce the complete decision process [ie the results of clicking on the + symbols under each decision, or just the decision itself].”

Because these users were working with an implemented guideline instantiated to a particular domain, they also referred to domain concepts including the following.

[3] “Adult/Child screen confusing, since ‘multiple choice’ option comes later”

[3] “Can’t see Family Ticket in ticket type selection”

[1] “Can’t do tickets in advance [e.g. for specified day which is not today]”

[3] “Machines [or the simulation] don’t tell you the cheaper route or choices, etc. ( the one offered may not be the most economical)”

[2] “Need clearer information on **ticket prices** on the [real] machines”

[3] “Tube map [on FFM] does not show where **zones** are, and zones [the concept and the boundaries] are confusing until you learn”

The set of domain concepts users worked with also included several from the task instruction sheet, and which any ticket purchaser works with (such as a ticket!), so these are also included in the CASSM model shown in Figure 4.

**Table 4:** Entities and attributes for Tallis Engine (TVM users)

	Concept	User	Interface	System	Set / Create	Change / Delete	Notes
E	guideline	present	difficult	present	fixed	fixed	difficult to get an overview of the guideline
A	evidence	present	present	present	easy	hard	
A	getting information	present	present	present	easy	hard	
A	intervention	difficult	present	present	easy	hard	"should be called 'candidates'"
E	ticket buying situation	present	notSure	notSure	fixed	fixed	
A	details	present	present	notSure	easy	hard	
E	plan	difficult	present	present	fixed	fixed	
E	action	difficult	present	present	fixed	fixed	
E	trigger	difficult	present	present	fixed	fixed	
E	decision process	difficult	difficult	present	fixed	fixed	
E	decision outcome	present	present	present	bySys	hard	
E	decision /data record	notSure	difficult	present	indirect	cant	
E	ticket	present	difficult	absent	fixed	fixed	
A	type	present	difficult	present	hard	notSure	
A	price	present	difficult	present	easy	notSure	finding cheapest ticket is hard
A	validity date	present	present	present	cant	cant	can only buy for today
E	train	present	absent	absent	fixed	fixed	
E	queue	present	present	notSure	fixed	fixed	
E	payment / money	present	present	notSure	fixed	fixed	
E	zone	present	difficult	present	hard	hard	

As in the case of the clinical Consumer, from the point of view of the end-user (Consumer) of the enacted TVMs guideline, much of what is made available is absent from the Consumer’s model (and cannot be switched off by the Consumer). In the view of the Composers and Consumers who were interviewed, these are Composer’s and not Consumer’s tools – a point made very explicitly by interviewee 2: “that’s not something I’d want my users - my end users - to see”.

### 3.3 Comparing the CASSM models

Tables 2, 3 and 4 can be compared against each other to establish the differences in models. A comparison of tables 3 and 4 supports understanding of how Tallis Engine can be used in different domains (in this case, clinical decision making and TVM use). More centrally to the theme of this paper, a comparison of tables 2 and 3 / 4 focuses attention on the C<sup>3</sup> chain, highlighting which concepts are transferred through the chain and which are not.

First, we briefly consider the differences between Tables 3 and 4. Essentially, the only difference between these tables is in the domain model. So patient information (presented sketchily in Table 3) is replaced by ticket-buying information in Table 4. The only other difference between these tables is the inclusion of the representation of evidence in Table 3 – included there because it was mentioned by one of the clinical interviewees but it did not emerge in any of the TVMs sessions.

More interesting is the difference between Table 2 and Tables 3/4. In this case, the important features are as follows:

1. Both Composers and Consumers reported difficulty in getting an overview of the guideline (although the role of an overview is different for the two user groups).
2. Consumers found it difficult to backtrack while running guidelines. This point did not emerge from the Composers' perspectives.
3. Domain information is absent from Table 2, because this is a generic decision support environment (albeit motivated by the requirements of clinical decision making). This has negative consequences for Consumers, who think in domain terms (e.g. "patient models") rather than decision processes.
4. Plans, Tasks, Triggers and PROforma are included in Tables 3 and 4, although this information is difficult for most Consumers to work with. Similarly, the representation of evidence is noted in Table 3 as being difficult for Consumers.
5. The decision outcome was noted by Consumers as being important; from a Composer perspective, this emerges through the interaction, and is therefore not an explicit concept.

The CASSM analysis of Tallis has highlighted both important differences and inappropriate overlaps between the Composer and Consumer models. Probably the two most important themes are the inappropriate emphasis on inspection of guideline components in the Engine (item 4 in the list above), and the focus on process rather than patient models (item 3).

The inclusion of Composer-relevant information in the Consumer system (item 4) suggests a conflation of the roles of Composers and Consumers, in that what is appropriate for the former has been assumed to also be of concern to the latter.

Conversely, the differences between the two user models is reflected in the differences of emphasis in the corresponding Cassata tables. In particular, a 'patient model' was found to be important for Consumers, and several Consumers expressed an interest in being able to backtrack through the decision process. A better understanding of Consumers' requirements might lead the developers to consider how to improve backtracking in the Engine environment, and whether to incorporate an explicit patient model within the Composer environment. Explicit inclusion of a patient model would make it more difficult to develop non-clinical guidelines, but could improve the 'fit' between the tool and the target context of use.

This illustrates how, for Tallis as for other composition tools, Creators need to be aware of both Composer and Consumer roles, while keeping them apart. In this particular case, in order to encourage clinicians who use guidelines to also create them, there may be a need for specific add-ons or enabling features which ‘upgrade’ from Consumer-level to Composer-level. However, this needs to be considered separately from the basic challenge of making such guidelines usable by and useful to clinicians in their every day work, without any expectation that all users will become guideline Composers.

#### 4. Discussion

We have shown how CASSM can be used to illuminate multiple classes of user model which form part of the ‘chain’ from designer to end user, and that tabulating results in the form demonstrated by Cassata enables the analyst to focus on the essential differences between these models. As discussed in the Introduction, the  $C^3$  chain is not specific to decision support or knowledge representation systems.

One role for CASSM in the development cycle is in pre-empting any conflation of Composer and Consumer models. CASSM does not explicitly differentiate between appropriate and inappropriate overlaps between models; a reasonable heuristic appears to be that Creators need to be more aware of the Consumer’s perspective, but that Consumers should not generally be expected to assimilate non-essential information about the Composer environment.

Elsewhere, we have compared the findings of CASSM analyses with those of procedurally based approaches such as Cognitive Walkthrough [6]. We have not conducted such a comparative analysis in the work with Tallis because, as should be evident from Figures 1 and 2, the procedures for working with the two interfaces are completely different. The Composer interface demands complex planning by users and an interaction based on a graphical drag-and-drop paradigm, whereas the Engine interface requires users to engage in a sequence of selections that leads them carefully through the decision process. The sequence embodied within the Engine interface is defined by the ordering of elements within the corresponding Composer knowledge representation, but is not reflected in the process that the Composer has to go through to construct the knowledge representation. These differences make it impossible to conduct a meaningful procedural comparison between the Composer and Engine interfaces; this contrasts with the conceptual comparison that CASSM has supported (section 3.3).

Tallis is an interesting example of the  $C^3$  model because decisions made by a Consumer at the early stages of an interaction session determine those aspects of the interface which will be available later on. Even website development tools may not expect this much premature commitment in the end product: at least with web sites one can backtrack and go down some different path, whereas Tallis does not offer such flexibility. However, Tallis may be unusual in having a ‘back-channel’ between Consumers and Composers, in that the same clinicians who make use of guidelines are also encouraged to compose them, and to upload them to the repository for others



to consume. In that sense, there may be a special benefit in the Consumer having a view of the Creator's world, in order to understand how the system has come to be.

Of course, programming support environments also expect Composers to act as Consumers when running, testing and debugging code, but it may be the special and detailed support for the interrogation of user outcomes (Table 4) that makes Tallis so prone to this kind of conflation. It is evident from the Consumer reports (Sections 3.2.3 and 3.2.4) that so much emphasis on intervention and diagnosis, rather than user control, can hinder rather than illuminate the support for outcomes.

CASSM can help to identify where in the 'chain' a particular tool is best used, because both Creators and Composers need comprehension of the other user models. In particular, Creators need to know about Composers and Consumers, and Composers need to know about Consumers. To what extent it is helpful for understanding to also flow the other way – that Consumers should understand the perspectives of Creators and Composers – remains an open question. Arguably, a ready-to-hand tool should not impose on its user the requirement to understand how it was made, or why it is the way it is. However, this is not the culture within which the Tallis development is taking place. In the current development context, the communications between the Creators, Composers and Consumers are perceived as being essential to the development of a shared culture of guideline development and use. However, the very culture that supports collaboration may also alienate potential Consumers who have no interest in being Composers. Such socio-political considerations are outside the scope of CASSM; nevertheless, the use of CASSM within this development culture has highlighted important questions about how information is presented to and used by different user populations.

## Acknowledgements

We are very grateful to all the participants in this study, both experts and novices, without whom this analysis would not have been possible, and to Paul Cairns for constructive criticism of a draft of this paper. The work on CASSM was funded by EPSRC (GR/R39108).

## References

1. Blandford, A., Green, T. & Connell, I. (2005) Formalising an understanding of user-system misfits. In R. Bastide, P. Palanque & J. Roth (Eds.) *Proc. EHCI-DSVIS 2004*. Springer: LNCS 3425. 253-270.
2. Blandford, A., Keith, S., Butterworth, R, Fields, B. & Furniss, D. (in press) Disrupting Digital Library Development with Scenario Informed Design. To appear in *Interacting with Computers*.
3. Blandford, A.E., Wong, B.L.W., Connell, I.W. and Green, T.R.G. (2002). Multiple viewpoints on computer supported team work: a case study on ambulance dispatch. In X. Faulkner, J. Finlay and F. Détienne (eds.), *People and Computers XVI. Proceedings of HCI 2002*, London, September 2002, pp. 139-156. London: Springer.

4. CASSM (2004) Shrinkwrapped tutorial, Cassata tool and worked examples. <http://www.ucl.ac.uk/annb/CASSM/>
5. Checkland, P. B. (1981) *Systems Theory, Systems Practice*. Chichester: John Wiley
6. Connell, I.W., Blandford, A.E. and Green, T.R.G. (2004). CASSM and Cognitive Walkthrough: usability issues with ticket vending machines. *Behaviour & Information Technology*, 2004, 23(5), 307-320.
7. Connell, I.W., Green, T.R.G. and Blandford, A.E. (2003). Ontological Sketch Models: highlighting user-system misfits. In E. O'Neill, P. Palanque and P. Johnson (eds.), *People and Computers XVII - Designing for Society*. Proceedings of HCI 2003, Bath, September 2003, pp. 163-178. London: Springer.
8. CRUK (2006a) Tallis system. <http://www.acl.icnet.uk/lab/tallis/>. Viewed 30/11/06.
9. CRUK (2006b) Tallis PROforma. [http://www.openclinical.org/gmm\\_proforma.html](http://www.openclinical.org/gmm_proforma.html). Viewed 30/11/06.
10. Fox, J., Beveridge, M. and Glasspool, D. (2003). Understanding intelligent agents: analysis and synthesis. *AI Communications*, 2003, 16, 139-152.
11. Fox, J., Johns, N. and Rahmzadeh, A. (1998). Disseminating medical knowledge: the PROforma approach. *Artificial Intelligence in Medicine*, 1998, 14, 157-181.
12. KPC (2006) Open Clinical Knowledge Publishing Collaboratory. <http://www.openclinical.org/kpc/Introduction.page>. Viewed 30/11/06
13. NetBeans (no date) <http://www.netbeans.org/> Viewed 12/02/07.
14. Nielsen, J. (1994) Heuristic Evaluation. In J. Nielsen & R. Mack (Eds.), *Usability Inspection Methods* (pp. 25-62) New York: John Wiley.
15. Norman, D.A. (1986). Cognitive engineering. In D.A. Norman and S.W. Draper (eds.), *User Centred System Design*, pp. 31-62. Hillsdale, NJ: Lawrence Erlbaum Associates.
16. Wharton, C., Rieman, J., Lewis, C. & Polson, P. (1994) The cognitive walkthrough method: A practitioner's guide. In J. Nielsen & R. Mack (Eds.), *Usability inspection methods* (pp. 105-140) New York: John Wiley.
17. Witten, I. H., Bainbridge, D. & Boddie, S. J (2001). Greenstone: Open-source digital library software with end-user collection building. *Online Information Review*, 25 (5), 288-298.