An Overview of Semiotic Engineering Epistemic Tools for the Design of Collaborative Systems

Raquel Oliveira Prates

Abstract Semiotic Engineering (2005) is an HCI theory that perceives an interactive system as a computer-mediated communication in which the designer of a system conveys to system users who the system is for, what it can be used for and how to interact with it. Based on this communicative perspective, the theory aims at providing explanation about the phenomena related to the design, evaluation, and use of interactive systems. To do so, Semiotic Engineering draws on *Semiotics* – the discipline that studies signs, significations processes and communication – and makes connections to Computer Science concepts.

Introduction

Within the communicative perspective of Semiotic Engineering, both designers and users are interlocutors of a communicative act, in which the designer is the sender of a message to users. The message is the interface itself, which conveys to users who are the users the system is designed for, what kind of tasks they can perform with the system and how they can interact with the system to achieve their goals. The content of the message can be paraphrased as:

> "Here is my understanding of who you are, what I've learned you want or need to do, in which preferred ways, and why. This is the system that I have therefore designed for you, and this is the way you can or should use it in order to fulfill a range of purposes that fall within this vision." (de Souza 2005; p. 25)

As users interact with the system, they understand the designer's message. Thus, the designer-to-user communication is in fact a meta-communication, since it takes place through the system-user communication. Also, it is unidirectional, since users do not have the opportunity to talk back to designers at interaction time.

R.O. Prates (✉)
Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, Brazil
e-mail: rprates@dcc.ufmg.br

S.D.J. Barbosa, K. Breitman (eds.), *Conversations Around Semiotic Engineering*, DOI 10.1007/978-3-319-56291-9_9

In this chapter, we focus on the Semiotic Engineering for collaborative systems. By collaborative systems we mean any system that aims to connect people to interact with each other through the system, be it to communicate, share information or coordinate activities (Grudin and Poltrock 2012), and thus encompasses a range of different systems, such as teleconference systems or social network sites.

In collaborative systems, although the designer's meta-message conveys the same overall decisions presented in the general template, this message becomes more complex since the designer is not communicating with one person at a time, but rather with a group of people (Prates and de Souza 1998). The designer's understanding of *who the users are* may need to consider the different roles they can take in the system, who can take each role, as well as the relationships among them. The definition of what they *want to do* and *how* may differ for each role. Furthermore, the interface must convey the designers' decisions not only about the users and the system, but also about how users can interact with other users through the system. Thus, the designer-to-user message must also transmit who can interact with whom; which codes and protocols are available for them to do so; and what information is available to them about others and their activities. Thus, in collaborative systems the designers' message to each role in the system may be different. Nonetheless, they must be consistent with each other. For instance, designers of a Virtual Learning Environment will have different messages to instructors and students about what they can do in the system, how to interact with it, what they can communicate with other users, and by which means. However, for the system to work well these different messages need to be consistent with each other and create an overall cohesive message about the system.

> Collaborative system meta-message template: "Here is my understanding of **who you are**, what I've learned **you want or need** to do, **in which preferred ways**, and **why**. This is the system that I have therefore designed for you, and this is the way you can or should use it in order to fulfill a range of purposes that fall within this vision. **You can communicate and interact with other users through the system.** To do so, the communication, the system will help you check:
>
> - Who is speaking? To whom? What is the speaker saying? Using which code and medium?
> - Are code and medium appropriate for the situation? Are there alternatives?
> - Is(are) the listener(s) receiving the message? What if not?
> - How can the listener(s) respond to the speaker? Is there recourse if the speaker realizes the listener(s) misunderstood the message? What is it?" (emphasis added) (de Souza 2005; p. 210).

In this chapter, we present our research along the years in supporting designers in the design of their meta-communication for collaborative systems. Thus, in the next section, we present some main Semiotic Engineering concepts that will be important to the understanding of this chapter. The following section presents the EpisTAMiCS architecture models. We then describe the existing models that have been proposed based on EpisTAMiCS. We end with a discussion of the status of the research in this direction, its contributions and next steps.

Some Semiotic Engineering Concepts in a Nutshell

Semiotic Engineering brings designers of interactive systems to the front stage of HCI processes and, ontologically, defines designers and users as equally important (de Souza 2005). By doing so, it aims to contribute to the awareness of what designers are doing and how it impacts users and their interaction with the system.

The designers message to users (i.e. the system's interface) is composed of signs. A *sign* is *anything that means something, in a context or situation, to somebody* (Peirce 1992–1998). Semiotic Engineering classifies the signs present in an interface into three types: *static*, *dynamic,* and *metalinguistic* (de Souza and Leitão 2009; de Souza et al. 2010). *Static* signs are interface signs that are present in the interface at any single moment in time and can be interpreted independently of causal and temporal relations (e.g. buttons, menus, images, etc.).

Dynamic signs are bound to temporal and causal relations of the interface, and can only be interpreted in relation to the interaction and behavior of the interface. For instance, in a text editor, as the user selects a word 'example' and presses the button that represents bold, the text changes to '**example**'. The dynamic sign is the behavior of changing the text to bold format. A dynamic sign usually represents the transition between two states of the system represented by static signs.

Finally, *metalinguistic* signs are interface signs that refer to other interface signs with the goal of explaining the meanings encoded in them. Typically, they are instructions, explanations or tips about other signs. They can be present at the interface level, such as tooltips or instructions, or in another context such as a help system or even a website about the system.

It is interesting to notice that an interface element may have static, dynamic, and metalinguistic signs associated to it. For instance, a button in a toolbar is itself a static sign in the interface. There may be a tooltip or an entry on the help system associated to it, which represent metalinguistic signs that explain it. Its behavior (what the system does when the user presses it) is a dynamic sign that associates two states of the interface represented by static signs.

The designer's activity in creating an interactive system involves deciding and producing the content and expression of the message. Defining the content involves learning or deciding who the target audience of the system is, what goals can be achieved by using it, and how to interact with it (interaction paradigms or principles to be used). Producing the expression of the message involves selecting and combining static, dynamic, and metalinguistic signs. The quality of the designer's message represents the system's *communicability* – that is, whether the system conveys to users the design intents and interactive principles in an organized and resourceful way, which allows them to achieve the intended result (Prates et al. 2000; de Souza and Leitão 2009).

Semiotic Engineering argues that any interactive system is a linguistic intellectual artifact (de Souza 2005 p. 10). A linguistic intellectual artifact is defined as the result of a human intellectual activity that encodes a particular interpretation of a problem, as well as of a particular set of solutions for that problem. The encoding is fundamentally linguistic, which means that it is encoded and can be interpreted based in a set of semantic rules. In order for the artifact to achieve its purpose, its users must understand the linguistic coding system in which it is formulated and be able to explore the available set of solutions proposed by its designer. Interactive systems also require that the linguistic system formulated be computable.

In order to support designers in designing their systems, Semiotic Engineering proposes that there should be tools available to support their intellectual activity. These tools should be *epistemic tools*, that is, tools that do not yield a direct answer to a problem, but support designers in understanding the problem itself, experimenting with different candidate solutions, evaluating the results, and anticipating the implications they may bring about.

In HCI, design models, scenarios, and guidelines can be perceived as epistemic tools, even if they are not framed as such. Nonetheless, the Semiotic Engineering epistemic tools differ from others since they are based on the perspective of the system as meta-communication artifact and support the designer in reflecting upon the system as a communicative act, even if different epistemic tools may focus on different aspects of the message and its design.

In the next section, we briefly present the main steps of a Semiotic Engineering design process, and describe a Semiotic Engineering based architecture model for collaborative systems epistemic tools that support designers in reflecting upon the content of the message they are sending to users.

Collaborative Systems Epistemic Tools

Throughout the years, some epistemic tools have been proposed within the Semiotic Engineering framework (Barbosa and Paula 2003; Silveira et al. 2001; Salgado et al. 2011; Barbosa et al. 2007). Existing epistemic tools have been proposed for different steps of the design process. In designing a communicative act, there are two main activities involved, defining *What to communicate?* (*content*) and *How to communicate it? (expression)*.

The first step in defining *What to communicate?* involves making the necessary decisions to fill out the meta-message template[1]. In other words, it involves deciding who the system is for, what they can do, and how to interact with it. The meta-message template itself can be perceived as an epistemic tool, which allows designers to express and reflect upon the overall message they want the system to convey. In the case of the collaborative system template, it may be easier to fill out a template for each role available, but they should be checked against each other to guarantee their cohesiveness.

[1] Notice that, although learning about the users, their needs and contexts would be expected, it is outside the scope of the Semiotic Engineering theory (de Souza and Leitão 2009). Such activities can be performed using known HCI methods, such as direct observation or interviews (Lazar et al. 2010).

Once designers have defined *what to communicate*, they can work on deciding *how to communicate it*. The expression of the message can be thought in terms of the interaction, in which the designer defines the possible system-user communications that can take place to convey the meta-message. To do so, a Modeling Language for Interaction as Communication (MoLIC) has been proposed (Barbosa and Paula 2003; Silva and Barbosa 2007). It was initially proposed for single-user systems, but it has recently been extended for collaborative systems (MoLICC) (Souza and Barbosa 2015; Souza et al. 2016).

In this chapter, we will focus on the epistemic tools aimed at supporting collaborative system designers in defining the content (or a specific part of it) of their meta-message. In this direction, some conceptual models have been proposed that aim at helping designers to focus, make decisions, and reflect upon part of the content of the message that they are defining. Although each proposed model has a different focus, they all stem from an overall support model. We will refer to it as EpisTAMiCS and will present it in the next section.

EpisTAMiCS Architecture Model

EpisTAMiCS stands for **Epis**temic **T**ools **A**rchitecture **M**odel for **C**ollaborative **S**ystems design. It proposes an architecture organizing components which could be useful to the proposal of epistemic tools aimed at supporting reflection on the content of the meta-message. The model is a review of the first architecture model proposed (Prates 1998) and its variants over the years (Barbosa 2006; Pereira Jr. 2016). The model is comprised of five components:

- **Design Language (DL)**: a design language that allows designers to describe (specific) aspects of the collaborative system model being designed. The lexical elements represent a set of dimensions that describe the collaborative system model in some aspect and the set of values each of these dimensions can take. The syntax of the language describes how the lexical elements can be combined and, its semantics, the possible meanings that can be expressed through the language.
- **Interpretive Rules**: a set of interpretive rules that act upon the model created by the designer using the design language. The set of rules identify possible combinations of the lexical elements of the DL and their values that could be problematic in some situations. The set of rules is defined as context-separable and descriptive (as opposed to being prescriptive) (Prates 1998). It is context-separable because the combinations identified do not take into account the actual context for which the system is being designed. It is left to the designers to take the context into consideration as they make decisions about the system. And it is descriptive because the rules should describe to designers the reasons why a specific combination has been identified as a potential problem in order to allow them to decide whether it is in fact an issue for the system being designed and its context, as well as to reflect upon other possibilities.

- **Design Rationale (DR)**: The Design Rationale component stores decisions regarding a model generated by the designer and the underlying reasons for them. Initially the DR component contains the explanation to all interpretive rules available. Whenever the designer complies with an interpretive rule that has pointed to a potential problem, the explanation for the rule represents the reason why the designer has made that decision. Ideally, the designer should be able to add information to the explanation to contextualize it, if necessary. In case the designer decides that a potential problem is not relevant, s/he should also be able to enter the explanation of why that issue is not a problem in that specific context. For a complete DR of a system, designers should be able to associate the rationale for their decisions to the model being created using the DL.
- **Future Scenario Generator**: One of the challenges in collaborative systems is being able to fulfill the needs of different people, different contexts, and even interaction styles among users through the system, as well as how they change over time (Prates et al. 2015a). Therefore, one popular solution is to offer users flexible systems, i.e. systems that may be customized or adapted to users or contexts. In order to do so, designers must define at design time what aspects of the systems can be defined at use time, by whom and how. Ideally, the DL would have lexical elements that allow designers to describe the flexible points of the system. Therefore, the goal of the Future Scenario Generator is to generate, based on the designers' description, the relevant scenarios that may be created by users at use time. This could be interesting for the designer, since the combination of aspects defined as flexible may allow users to create a large number of different scenarios, and they may not be easily anticipated by the designer. Also, by applying the interpretive rules to them, designers could be able to identify any scenarios that may be problematic or undesirable to users.
- **Design Description Interpreter**: Analyzes the model created by the designer using the DL. It identifies lexical or syntactic problems, and also checks if any of the interpretative rules are being broken, and if so points it out to the designer.

Figure 1 depicts the EpisTAMiCS model and how it would be used by a designer. The designer would use the *design language* available to describe (part of) the content of meta-message, in other words, to create a conceptual model of (part of) the collaborative system being proposed. Her/his explanations about the model created would go into the *design rationale* base. If the designer had described any changes that could take place in time in the system, the *future scenario generator* would generate the description in the design language for the (relevant) scenarios that could be created. The *design interpreter* would check the model (description of system and future scenarios). To do so, it would check the lexical and syntactic description of the model, and then apply the *interpretive rules*. In case the rules identified any relevant situation, it would present to the designers the situation identified and the explanation about potential problems or aspects that designers should take into consideration. Designers could then change their models to comply with rules or could enter into the *design rationale* base an explanation why in the context of the system the situation was not a problem. Notice that in Fig. 1 we depicted the

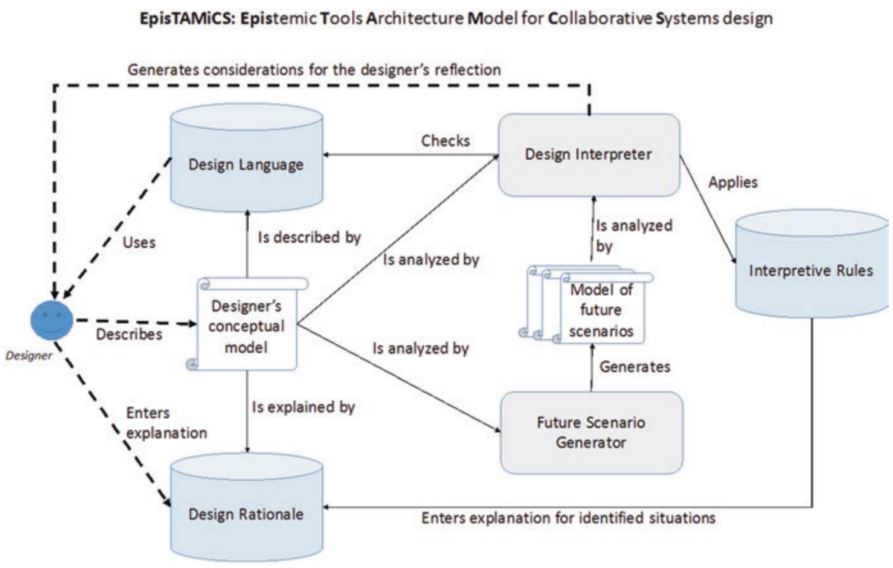

EpisTAMiCS: Epistemic Tools Architecture Model for Collaborative Systems design
Generates considerations for the designer's reflection
Design Language
Checks
Design Interpreter
Applies
Uses
Is described by
Is analyzed by
Is analyzed by
Interpretive Rules
Model of future scenarios
Designer's conceptual model
Describes
Designer
Is analyzed by
Generates
Enters explanation
Is explained by
Future Scenario Generator
Design Rationale
Enters explanation for identified situations

Fig. 1 EpisTAMiCS: Epistemic tools architecture model for collaborative systems design

interactions between the designer and the EpisTAMiCS components using dashed lines.

EpisTAMiCS family models intend to support designers of collaborative systems in two different capacities. The first one is by offering the DL to describe (part of) their meta-message. By doing so, the designer has to reflect upon the dimensions represented in the DL, the possible values they could take, and which one represents their intended solution. Representations in general work as epistemic tools since the definition of what to express and how to do so tend to lead to the identification of aspects that may not have been thoroughly defined yet or that might be inconsistent with other aspects represented. The difference of the Semiotic Engineering epistemic tools is that they are based on the theory and its perspective of the interface as a communicative act, leading the designer to reflect upon it explicitly or implicitly. The second capacity is by offering designers the interpretative set of rules, which can identify points of the model defined that might raise a potential issue, and explaining to designers what these are. This will allow designers to take into account potential aspects that could be problematic in some contexts and to analyze whether it would be the case in the context of the system being developed.

Next we will briefly describe the models that have been proposed within the EpisTAMiCS framework. We will notice that most of them have not defined all the components of the architecture model (yet), but having been conceived within the context of EpisTAMiCS, one may consider their addition in the future.

EpisTAMiCS Family Models

MArq-G

The first model of the EpisTAMiCS family was MArq-G – its architecture model contained all the components described above[2]. However, the Future Scenario Generator was only described theoretically and not instantiated (Prates 1998; Prates and de Souza 1998). The design language in the model was denominated MetaCom-G, and it focused on group models and allowed designers to describe the main features of a groupware system. The lexical elements of the DL described: the *roles* the users can take; the *hierarchical* relation among them; the group's *collaboration model* – i.e. how much each role's activity depended on the activity of other members; the *objects* available in the system, and whether they were private or shared (and by whom); with whom each member could *communicate* and how, and what they could *see* in the system. The set of rules contained heuristics that aimed at identifying situations that could hinder the group's work, such as if users shared an object but could not communicate to each other about it; or if members who collaborated with each other by having to do some activities together did not have any awareness of the other members' activities. The Design Rationale base contained the explanation for the rules and allowed designers to enter an explanation whenever a potential problem was identified which s/he did not consider to be a problem in the context of the system.

Later these models were extended to provide the possibility of a richer description of the communication among group members (Barbosa et al. 2005). The motivation for this extension was that MetaCom-G focused on the collaboration among members based on their activities, whereas in some systems, such as online communities, communication itself was the main activity among members. The new DL, denominated MetaCom-G*, allowed for a more detailed description of the communication among group members. To do so, it exchanged the existing lexical elements that allowed designers to express that there was a communication between members for lexical elements that allowed designers to describe the purpose of the communication based on Searle's Speech Act Theory (Searle 1992) and its structure.

Prototypes to allow for the use of these models (original and extended versions) as epistemic tools were developed in Prolog, and had as interface a command language that implemented the model's design language. Both models underwent preliminary evaluations, conducted mainly by their proponents.

[2]This first model proposed an extra component, which was the Widget Advisor. The goal of the Widget advisor was to suggest possible interface elements, based on the model described. However, this component was later dropped, since the architecture model aimed at supporting designers in defining the content of the meta-message and the widget advisor was an attempt to support the designer in thinking about its expression. Furthermore, the other components focused on an abstract model, whereas the Widget Advisor would be dependent on specific technologies.

Manas

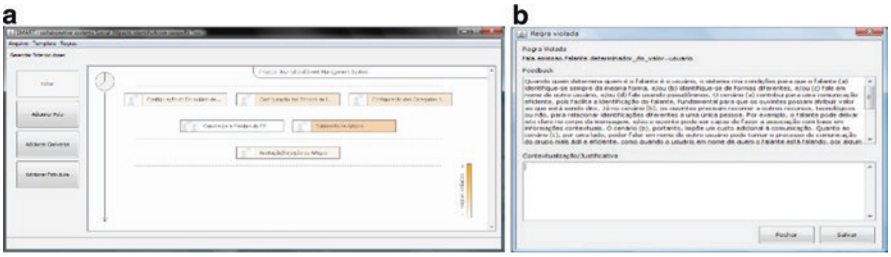
Once Semiotic Engineering was consolidated as a theory (de Souza 2005), there was a new proposal of a model based on EpisTAMiCS, denominated Manas (Barbosa 2006; Barbosa et al. 2007). The goal was that the model would help designers to not only make decisions about the collaborative activity, but also lead them to think explicitly about the design as a communicative act. To achieve this goal, Manas proposed describing any interaction among users through the system as a communicative act among them. Manas' architecture model was based on EpisTAMiCS, except for the Future Scenario Generator, which was not included. The design language proposed, named L-ComUSU, proposed as lexical elements: *interlocutors* (senders and receivers – addressed and not-addressed), the *purpose of the communication* (described by Searle's speech act), its *topic* and *content*. For each of these elements a set of attributes should be defined. Namely, designers should describe if there was an *explicit representation* of the value of the dimension within the system; what its *scope* would be (or value it could take); who would *determine the value* (user or system), if the value should be *mandatory*, and if a *default value* should be assigned. Also, designers should specify what level of processing the system should perform on the information (e.g. just show it, organize it or generate new aggregate data).

Manas's interpretive rules focused on potential social impacts of the systems upon users. Thus, the heuristic rules proposed identified aspects related to social values, such as politeness and privacy. Often the rules would indicate benefits and costs associated to a decision (and not only potential problems). For instance, if the designer decided to explicitly represent in the system the purpose of a communication, Manas would inform her/him that, if on the one hand this decision could provide for a clearer communication intent, but on the other hand it might not be pleasant to the listener or even put the speaker in an awkward position. Another example would be if the designer defined that the system would be responsible for determining who the receivers of a message were, Manas would inform her/him that this could be efficient to maintain communication protocols, but could prevent senders from privately speaking to other users, which might not be desirable.

Manas was evaluated in an analytical context and a review of some of the lexical elements was proposed (da Silva and Prates 2008; da Silva 2009). The review did not introduce any new elements, but proposed changes to some of the existing ones. The main change was to propose that for any user communicative act it would be relevant to be able to represent its attributes for senders and receivers separately (if needed), since they might have a different view of some of the elements of the communicative act. For instance, in sending a message it might be possible for the sender to include not-addressed receivers, of whom the addressed receivers would not be aware. Also, a new attribute was included that allowed designers to specify in which *moment* of the communicative act the element would be explicitly represented (during the definition of the message, or after it had been sent). Furthermore, changes to the set of values of two other dimensions were proposed (explicit representation: to allow for the definition of which type of signs would be used in the representation – metalinguistic, dynamic or static; and values for processing level). A few new rules related to the changes and new issues that might be relevant were added to the original set of rules.

Based on this review, a modeling tool for Manas – SMART (collaborative systems **S**ocial i**M**pacts identific**A**tion suppo**R**t **T**ool) – was developed and made available (Fig. 2 shows some SMART screenshots) (da Silva et al. 2010). The tool allowed other analyses of Manas in a design context (as part of a course's project) (Prates and Silva 2010), as well as other evaluation contexts (da Silva and Prates 2008; Barros and Prates 2014). The main findings of these works were that Manas was useful in designing and evaluating collaborative systems, and supported designers in reflecting upon the system's social impacts. In the design experiment one of the participants commented that by using Manas L-ComUSU, they paid more attention and thought more about social aspects of their design than they had up to that moment. Some participants also reported that some of the explanations about potential problems led them to make changes to their designs. The evaluation of existing systems using Manas allowed evaluators to identify points they considered to be problems and to think about redesign proposals for the system. These works reported that the main cost in using Manas was learning L-ComUSU (what the dimensions meant and their values), as well as the theory and knowledge that was necessary to understand Manas and use it, for instance, how to think about and express activities in the system as communicative acts.

More recently, new models have been proposed with focus on specific problems – one of them focuses on helping designers anticipate possible scenarios resulting from user configurations and the other on privacy in social network sites. In both cases, one point considered was whether they should extend one of the existing models, or should propose yet a new model and design language. In both works, the decision was to propose a new model. The main cost of this decision is having many different models available that do not interact with each other, making it dif-



Quando quem determina quem é o falante é o usuário, o sistema cria condições para que o falante (a) identifique-se sempre da mesma forma, e/ou (b) identifique-se de formas diferentes, e/ou (c) fale em nome de outro usuário, e/ou (d) fale usando pseudônimos. O cenário (a) contribui para uma comunicação eficiente, pois facilita a identificação do falante, fundamental para que os ouvintes possam atribuir valor ao que está sendo dito. Já no cenário (b), os ouvintes precisam recorrer a outros recursos, tecnológicos ou não, para relacionar identificações diferentes a uma única pessoa. Por exemplo, o falante pode deixar isto claro no corpo da mensagem, e/ou o ouvinte pode ser capaz de fazer a associação com base em informações contextuais. O cenário (b), portanto, impõe um custo adicional à comunicação. Quanto ao cenário (c), por um lado, poder falar em nome de outro usuário pode tornar o processo de comunicação do grupo mais ágil e eficiente, como quando o usuário em nome de quem o falante está falando, por algum

Fig. 2 SMART (Interface in Portuguese). (**a**) Presents each communicative act and how they are structured; the color represents whether there is any feedback regarding the rules associated to that communicative act; (**b**) When a rule is broken, it presents the rule and its explanation to the designer, and allows him/her to include their explanation about the context

ficult to provide a suite that could allow designers to benefit from having learned one, when using the others. Also, in terms of Semiotic Engineering theory, a more cohesive set of models could be more useful in allowing for a broader adoption of the theory and its models. Nonetheless, one aspect that motivated the decision was that the existing models could not easily represent the dimensions identified as relevant. Thus, it would require a large adaptation of the design language, and it could make learning the models, which is costly, even more so. Furthermore, since the focus was very specific, having a more focused model could be more attractive to designers that could be potential users of the models.

SIGMa

Allowing users to customize collaborative systems to their own needs and contexts makes systems more flexible, and allows them to better fulfill users' needs in different situations. In order to allow collaborative systems to be customizable, designers must define, at design time, all the features or parameters that should be flexible, and how users can change them at use time. One of the challenges of designing flexible systems is that designers may not be able to anticipate all the possible scenarios that users may create by changing and combining flexible aspects of the system (Prates et al. 2015a). Thus, designers may inadvertently make it possible for users to generate scenarios that would not be desirable (Pereira Jr. et al. 2014).

In order to support designers in modeling how users' interaction with the system and among themselves through the system could change over time, SIGMa (**S**cenar**I**o **G**enerator **M**odel) was proposed. It was based on EpisTAMiCS, with special focus on the Future Scenario Generator component, that before SIGMa had only been described theoretically, but had not been instantiated in a working prototype. It was decided that a new design language that focused on possible changes would be better to investigate the full potential of such a model[3]. The goal was to create a language that could express well different types of changes that designers may want to make available to users in a concise way. The focus was the *design language* and *future scenario generator*. The *interpretative rules* component was planned to be focused on at a later moment, and only a few rules were identified as a proof of concept.

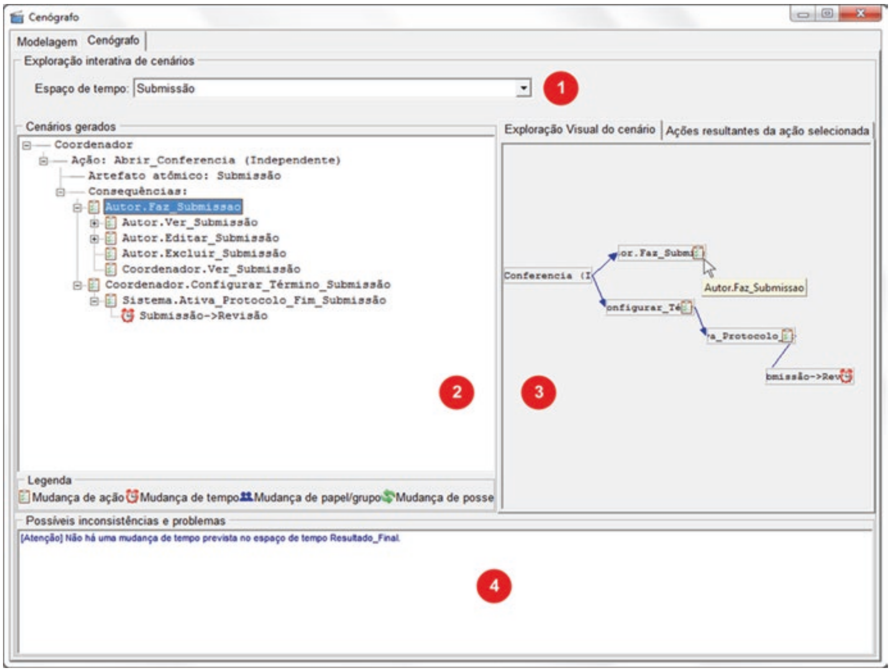
SIGMa-dl contains as lexical elements: time period, roles, groups, artifacts (unary or composed), ownership, actions, changes, and relations. *Time period* rep-

[3] The alternative would be to use MetaCom-G or Manas L-ComUSU. Although MetaCom-G had the concept of expressing changes over time, they were limited. Furthermore, since MetaCom-G was not being used, there would be no requirement or need to maintain it and try to adapt it. Manas, on the other hand, was more recent and had been used more. However, it did not include the concept of time and changes, and including that would require a large change in the design language and might make it more complex and less usable.

resents a period of time in which context within the system does not change (and not a chronological time). *Roles* describe the roles that users can take in the system, whereas *groups* allows designers to describe a set of behaviors available to a group that includes members of different roles, or a subgroup of members of a role. *Artifacts* describe elements that can be manipulated by users within the system – unary artifacts are a low-level unit that makes sense; whereas composed artifacts are those that combine unary or other composed artifacts. *Ownership* allows designers to define which artifacts are owned by which roles or groups. *Actions* represent the description (conceptual, not as a sequence of steps) of possible actions in the system that different roles or groups can take. Finally, *changes* and *relations* are deeply related. *Changes* describe what possible changes the designers want to make available in the system. There are four types of changes that can be described: role change, change of the time period (or context of the system), changes in an artifact, and changes in the set of actions available. *Relations* are how designers can associate a described change to an action that may trigger it.

To model a system using SIGMa-dl, the designer must define at least one time period and describe the system using the other lexical elements available. The goal of the Scenario Generator component (denominated SIGN) is to allow designers to explore the future scenarios, by simulating which changes may come into effect under which circumstances and analyzing how the context of the system will change in time. A prototype that supports designers in modeling possible changes using SIGMa-dl and exploring possible changes and their impacts on the systems context has been implemented (Fig. 3).

The model has been evaluated in three different aspects. First of all, it was evaluated using the Cognitive Dimensions of Notations (CDN) – a framework that provides a vocabulary that allows for the evaluation of notations (Blackwell and Green 2003). Next, in order to evaluate its expressiveness, SIGMa-dl was used to describe existing systems that allow users to customize aspects of the system – namely Facebook and Google Inactive Account Manager. The goal was to use it to describe real customization possibilities and analyze whether all the systems' possibilities could be described using SIGMa-dl, as well as inspect whether the scenarios generated by SIGN were consistent with the ones that could actually be created in the systems. In special, in a previous work, a problematic scenario for users of Facebook was pointed out (Pereira Jr. et al. 2014), and it was relevant to analyze if it would be correctly generated by SIGN. Finally, SIGMa was evaluated with six system designers who represented its potential users. The evaluation consisted in an activity that involved the analysis of an existing system, and a (partial) modeling of a new system. The results of these evaluations generated positive indicators about SIGMa's expressiveness (there were no aspects or situations of the systems considered in the evaluations that SIGMa-dl could not identify); and its role as an epistemic tool (the use of SIGMa in the analysis and design activities led participants to change their views of the system or on how to design it) (Pereira Jr. 2016).



Espaço de tempo: Submissão

```
Coordenador
  Ação: Abrir_Conferencia (Independente)
    Artefato atômico: Submissão
    Consequências:
      Autor.Faz_Submissao
        Autor.Ver_Submissão
        Autor.Editar_Submissão
        Autor.Excluir_Submissão
        Coordenador.Ver_Submissão
      Coordenador.Configurar_Término_Submissão
        Sistema.Ativa_Protocolo_Fim_Submissão
          Submissão->Revisão
```

[Atenção] Não há uma mudança de tempo prevista no espaço de tempo Resultado_Final

Fig. 3 SIGMa prototype (in Portuguese). There are two tabs, one for modeling and one for the scenario generator (being depicted): (1) indicates the time period being explored; (2) shows the scenario generated for that time period; (3) shows a graphical view of scenarios; (4) shows syntactic or semantic inconsistencies identified

Privacy Design Model (PDM)

The final model that has been based on EpisTAMiCS is the Privacy Design Model (PDM), which focuses on supporting designers in modeling privacy aspects regarding users' communication through social network sites. With the wide adoption of collaborative systems, in special social network sites, privacy became a "hot topic". Although there is a broad corpus of research on privacy, one challenge that remains open is to support designers' in incorporating privacy principles at design time (privacy by design) rather than as an afterthought (Cavoukian 2006).

PDM draws not only on Semiotic Engineering, but also on Altman's theory of privacy (Altman 1975). PDM structures the user-system-user communication based on Semiotic Engineering's design space (Villela and Prates 2015; Villela 2016). Thus, PDM's design language requires the designer to reflect upon user to user communication through the system and their impacts and represent it in terms of: *information source* (who is the sender of the information being shared about an individual[4]); *audience* (who is the receiver of the information being shared);

[4] In this chapter, we refer to the user about whom the information is being shared as *individual*.

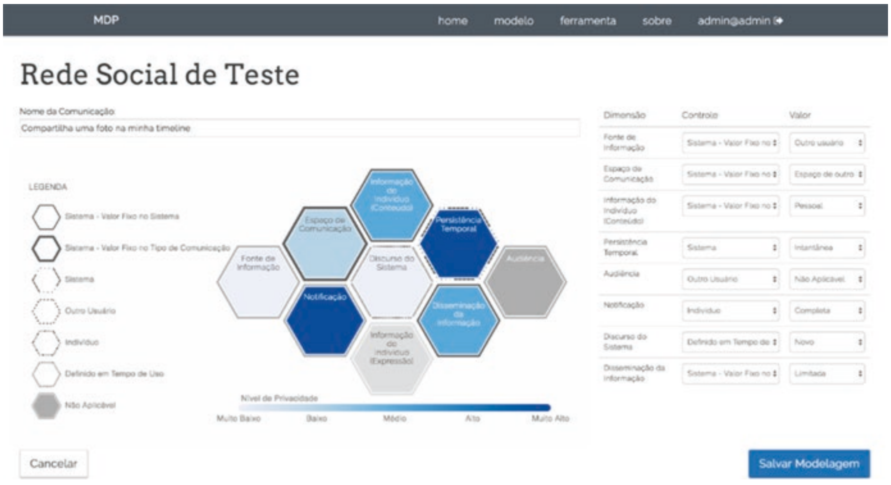
communication space (where within the system the individual's information is being disclosed); *individual's information expression* (whether the format in which the information is expressed in the system is predefined by the system or defined by the user); *individual's information content* (defines how personal is the individual's information being shared); *temporal persistence* (defines how long the information being shared is available within the system for). There are also lexical elements that refer to the effects of the communication within the system (e.g. how it may be disseminated by the system or other users): *notification to individual* (describes whether the system informs the individual when other users perform any activities involving his/her information); *speech about the individual* (describes whether the system is able to disclose information about an individual to other users on its own); *information dissemination* (describes whether the audience is able to share an individual's information).

The designer must identify each possible type of user-to-user communication in which personal information can be shared, and then describe it using PDM. For each of the dimensions above, the designer chooses one or more values the dimension can take. For each dimension designers also define who has control over its value, in other words, who can assign its value: if the designer – either at design time or through the system at use time – or the user. If the value of the dimension is left for users to define, designers define at design time the set of possible values that users can assign to it. As of now, PDM contains only the DL component of EpisTAMiCS. There is also available to designers a discussion on how different combinations of values for the dimensions increase or decrease users' privacy in the system (Villela 2016). Nonetheless, this knowledge has not yet been formalized in terms of rules that could be used to compose the *Interpretative Rule* component. An investigation of whether this knowledge could be organized in terms of heuristic rules to be applied on the model created by the designer is a future step of the research.

A visual representation of the model has been proposed aimed at providing designers with an overview of the different privacy dimensions related to a possible communication type between users. Based on this visual representation a tool – PryMeVis[5] (**PR**ivac**Y** design **M**odel **Vis**ualization) – has been developed to support the use of PDM (Villela et al. 2016) (see Fig. 4). The visual representation depicts a honeycomb and each PDM dimension is shown as a hexagon. PryMeVis requires designers to define the relevant communication types in the system and for each one of them to define the value to be assigned to each dimension, as well as determining who controls it. The hexagons are filled with a color that is related to the possible set of values – the darker the color, the higher the level of privacy assigned to the dimension. Figure 4 shows a screenshot of PryMeVis of a communication type described by the designer.

An initial evaluation of PDM was performed with the goal to collect qualitative indicators on the PDM DL's expressivity (Villela 2016; Villela and Prates 2016). In that direction, two aspects were of interest: (1) whether PDM was able to describe

[5] Available at: http://pensi.dcc.ufmg.br/applications/



Rede Social de Teste

Nome da Comunicação: Compartilha uma foto na minha timeline

Dimensão	Controle	Valor
Fonte de Informação	Sistema - Valor Fixo no	Outro usuário
Espaço de Comunicação	Sistema - Valor Fixo no	Espaço de outro
Informação do Indivíduo (Conteúdo)	Sistema - Valor Fixo no	Pessoal
Persistência Temporal	Sistema	Intantânea
Audiência	Outro Usuário	Não Aplicável
Notificação	Indivíduo	Completa
Discurso do Sistema	Definido em Tempo de	Novo
Disseminação da Informação	Sistema - Valor Fixo no	Limitada

Fig. 4 PryMeVis – Visual tool for PDM modeling (in Portuguese)

privacy decisions in Social Network Sites (SNSs); and (2) whether it could express the differences in the designer's decisions about privacy. In order to explore these issues, PDM was used to describe three different existing SNSs (Facebook, ResearchGate and CaringBridge) and in an analytical activity with potential users of the model. The results indicated that PDM DL could be used to describe the different privacy aspects of the SNSs. Furthermore, through its representation of the design decisions, designers could understand their differences and discuss how different combinations could be more interesting in the distinct contexts – showing its potential as an epistemic tool.

Discussion and Final Remarks

All the models of the EpisTAMiCS family aim at supporting designers in defining part of the content of their collaborative system meta-message. In other words, the goal is to support them in taking into consideration relevant aspects of a collaborative system as they are making decisions about the solution they are going to offer users. The models' design language lead designers to think about the different dimensions and their values and combinations as they use it to describe their solution. The models interpretative rules go further and point designers to situations that could result from their decisions that may be potential problems, and worth considering.

As we have presented, MetaCom-G focuses on an overall description of activities of groups working together, and the impact of the designer decisions may have on the interactions and relations of group members. SIGMa-dl dimensions also allow for the description of group activities (allows for the representation of objects, with whom they are shared and how), but focus mainly on allowing designers to describe changes and generating the future scenarios for designers' analyses. SIGMa was motivated by the original MArq-G work and efforts to create systems that can be customized, adapted or extended by users. Even in looking only at customizable systems there are a number of challenges that users face (Pereira Jr. et al. 2014; de Souza et al. 2010; Prates et al. 2015b, 2016), and that designers should consider and anticipate at design time (Prates et al. 2015a). Nonetheless, there is not much work that supports designers in this effort (Wulf and Golombek 2001), and SIGMa is an epistemic tool that may be useful in advancing the state of the art in this direction.

Manas and PDM focus more on social aspects of systems' use. Manas allows for the description of the communication among group members and analyzes (by means of its heuristic rules) the social impacts it may have on users. It is interesting to note that Manas can be used to model systems in which there is no direct communication, by thinking and modeling activities as indirect communicative acts (da Silva and Prates 2008; da Silva 2009). By doing so, it makes the Semiotic Engineering communication perspective explicit to designers and helps them frame their decisions as communicative acts. Although Manas is able to offer some feedback regarding privacy, it does not make explicit to designers which aspects of the communication directly impact users' privacy. As current systems and technologies have made privacy a relevant concern for users and designers, PDM was proposed to support designers in taking privacy into account at design time for social network sites. It frames communication among users in terms of the Semiotic Engineering design space dimensions and associates them to privacy aspects.

Learning a new representation has an associated cost to designers. However, in order for the model to be useful, its benefits need to justify the cost of learning it. Although all the EpisTAMiCS family models have been initially evaluated, most of the assessment performed focused on the expressiveness of the model and identifying whether it could bring benefits as epistemic tools, that is, whether they could lead designers to take into consideration new relevant aspects and situations they had not reflected upon. Models that have a very specific focus, such as SIGMa and PDM, can have their use motivated in contexts in which future scenarios generated by customization and privacy are relevant issues to designers. The downside is that there are fewer opportunities to use the design language learned and even consolidate its understanding, if compared with more general models such as ConcurTaskTrees (CTT) (Paternò 2004) or UML (Rumbaugh et al. 2004). At any rate, they bring a relevant contribution to the investigation on how to deal with these issues at design time.

All the EpisTAMiCS family models advance the research in Semiotic Engineering by investigating tools that can support a design process that is based on the theory, even if they currently only support part of process needed to make decisions and design a meta-communication act. Furthermore, by increasing the number of case studies of systems being designed within a Semiotic Engineering theoretical grounding, we can collect data that in the long term will allow us to discuss how theoretical based models, methods, and approaches can or cannot offer different results than empirical HCI methods.

The next steps in this research involve performing a more extensive evaluation of the models including collecting data on each one's costs and benefits. Also, a direction of interest is investigating how these models can be integrated with other Semiotic Engineering epistemic tools, such as MoLICC. The idea would be not only to combine them in different steps of the process, but rather to explore whether one could facilitate in any capacity the generation of the next one.

References

Altman I (1975) The environment and social behavior: privacy, personal space, territory and crowding. Brooks/Cole Pub. Co

Barbosa CMA (2006) Manas: an epistemic tool to support the design of communication in collaborative systems (in Portuguese). DSc. thesis – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro

Barbosa SDJ and Paula MG (2003) Designing and evaluating interaction as conversation: a modeling language based on semiotic engineering. International Workshop on Design, Specification, and Verification of Interactive Systems, Springer

Barbosa CMA, Prates RO and de Souza CS (2005) MArq-G*: a semiotic engineering approach for supporting the design of multi-user applications. Proceedings of the 2005 Latin American conference on Human-computer interaction, 128–138

Barbosa CMA, Prates RO and de Souza CS (2007) Identifying potential social impact of collaborative systems at design time. In: Proceedings of INTERACT, Springer, 31–44

Barros EFM, and Prates RO (2014) Uma análise comparativa dos modelos de sistemas colaborativos fundamentados na engenharia semiótica. In: Proceedings of the 13th Brazilian symposium on human factors in computing systems. Sociedade Brasileira de Computação

Blackwell A and Green T (2003) Notational systems – the cognitive dimensions of notations framework. In: HCI models, theories, and frameworks: toward an interdisciplinary science. Morgan Kaufmann

Cavoukian A (2006) Privacy by design the 7 foundational principles implementation and mapping of fair information practices

da Silva RF (2009). ManasTool: Uma ferramenta computacional para apoio ao projeto da comunicação entre usuários em sistemas colaborativos. Dissertação de mestrado, Universidade Federal de Minas Gerais

da Silva RF and Prates RO (2008) Avaliação da Manas na identificação de problemas de impacto social: um estudo de caso. In: IHC '08: Proceedings of the VIII Brazilian symposium on human factors in computing systems. Sociedade Brasileira de Computação, pp 70–79

da Silva RF, Prates RO and Salles VJ (2010). Smart: a computational tool for supporting the identification at design time of the social impact of collaborative systems. Anais do Simpósio Brasileiro de Sistemas Colaborativos, pp. 39–46

de Souza CS (2005) The semiotic engineering of human-computer interaction. MIT Press, Cambridge, MA

de Souza CS, Leitão CF (2009) Semiotic engineering methods for scientific research in HCI. Morgan & Claypool, Princeton

de Souza CS, Leitão CF, Prates RO, Amélia Bim S, da Silva EJ (2010) Can inspection methods generate valid new knowledge in HCI? the case of semiotic inspection. Int J Hum Comp Stud 68(1-2):22–40

Grudin J, Poltrock S (2012) CSCW: Computer supported cooperative work. Encyclopedia of human-computer interaction. Aarhus, The Interaction Design Foundation

Lazar J, Feng JH, Hochheiser H (2010) Research methods in human-computer interaction. Wiley, New York

Paternò F (2004) ConcurTaskTrees: an engineered notation for task models. The handbook of task analysis for human-computer interaction, pp 483–503

Peirce, C. (1992–1998). The essential Peirce (Vols. 1 & 2). In N. Houser and C. Kloesel (eds.) The Peirce edition project. Bloomington: Indiana University Press.

Pereira Junior M (2016) Um Modelo para Apoiar Projetistas de Sistemas Colaborativos na Antecipação de Cenários. PhD thesis, Departamento de Ciencia da Computacao, UFMG

Pereira Jr M, Xavier SIR, and Prates RO (2014) Investigating the use of a simulator to support users in anticipating impact of privacy settings in Facebook. In: Proceedings of the 18th ACM international conference on supporting group work, ACM, 63–72. http://doi.org/10.1145/2660398.2660419

Prates RO (1998) The semiotic engineering of multi-user interface languages (in Portuguese). DSc. thesis – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro

Prates RO and de Souza CS (1998) Towards a semiotic environment for supporting the development of multi-user interfaces. In: Proceedings of international conference on collaboration and technology (CRIWG 1998), 53–67

Prates RO, and Silva RF (2010) Avaliação do uso da Manas como ferramenta epistêmica no projeto de sistemas colaborativos. In: Proceedings of the IX symposium on human factors in computing systems. Brazilian Computer Society

Prates RO, de Souza CS, Barbosa SDJ (2000) A method for evaluating the communicability of user interfaces. ACM Interac 7(1):31–38

Prates RO, Rosson MB, de Souza CS (2015a) Interaction anticipation: communicating impacts of groupware configuration settings to users. End-user development, Springer, 192–197

Prates RO, Rosson MB, de Souza CS (2015b) Making decisions about digital legacy with Google's inactive account manager. In: Proceedings of INTERACT 2015, 6 pp

Prates RO, Rosson MB, de Souza CS (2016) Analysis of configuration decision space over time: the Google inactive manager account case. In: Proceedings of XV Brazilian symposium on human factors in computer systems (IHC 2016), Sociedade Brasileira de Computação

Rumbaugh J, Jacobson I, Booch G (2004) The unified modeling language reference manual, Pearson Higher Education

Salgado LCC, de Souza CS, Leitão CF (2011). On the epistemic nature of cultural viewpoint metaphors. In: Proceedings of the 10th Brazilian symposium on human factors in computing systems and the 5th Latin American conference on human-computer interaction, Brazilian Computer Society, 23–32

Searle JR (1992) Conversation reconsidered. In: Parret H, Verschueren J (eds) (on) Searle on conversation. John Benjamins, Amsterdam, pp 137–148

Silva BS, Barbosa SDJ (2007) Designing human-computer interaction with MoLIC diagrams – a practical guide. In: Monografias em Ciência da Computação. Rio de Janeiro, PUC-Rio

Silveira MS, de Souza CS, Barbosa SDJ (2001) Semiotic engineering contributions for designing online help systems. In: Proceedings of the 19th annual international conference on computer documentation. ACM

Souza LG, Barbosa SDJ (2015) Extending MoLIC for collaborative systems design. In: Proceedings of the 17th HCI international conference. Springer, Los Angeles, pp 271–282

Souza LG, Barbosa SDJ, Fuks H (2016) Evaluating the expressiveness of MoLICC to model the HCI of collaborative systems. In: International conference of design, user experience, and usability. Springer

Villela MLB (2016). Um Modelo de Design de Privacidade para o Compartilhamento de Informacoes Pessoais em Redes Sociais Online. PhD thesis, Departamento de Ciencia da Computacao, UFMG

Villela MLB, Prates RO (2015) Supporting designer in modeling privacy for social network sites. In: Proceedings of XIV Brazilian symposium on human factors in computer systems (IHC 2015), Sociedade Brasileira de Computação, 113–122

Villela MLB, Prates RO (2016) Privacy design model for social network sites. 27 p (Submitted)

Villela MLB, Ferreira LS, Barros DAdF, Prates RO, Melo-Minardi, RCD (2016) PryMeVis: Uma ferramenta para modelagem de design de privacidade. In: Proceedings of XIII Brazilian Symposium of Collaborative Systems (SBSC 2016), Porto Alegre – RS. Sociedade Brasileira de Computação

Volker Wulf, Björn Golombek (2001) Exploration environments: concept and empirical evaluation, Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work, September 30-October 03, 2001, Boulder, Colorado. doi:10.1145/500286.500304