What's in a User Story: IS Development Methods as Communication

Rodney J. Clarke *rclarke@uow.edu.au*
Faculty of Business, University of Wollongong
Wollongong, Australia

Karlheinz Kautz *kautz@uow.edu.au*
Faculty of Business, University of Wollongong
Wollongong, Australia

Abstract

This paper challenges claims made by Scrum proponents when characterising the communicative nature of user stories: including being more 'authentic' because they comprise spoken language and that they are stories. We argue and decisively demonstrate that neither can be upheld. By incorrectly characterising user stories, we miss opportunities to understand what they are and how they work during development. User stories are better understood by applying a functional theory of communication that emphasises how language is used. By selecting systemic functional linguistics, we can analyse user stories, and have developed a method for factoring unwanted epics into usable user stories.
Keywords: User Stories, Epic, Scrum, Systemic Functional Linguistics, Thematic Progression.

1. Introduction

This paper concerns the nature of user stories within Agile Scrum- a commonly used management method for incremental product and service development [13,18]. User stories have a very specific canonical form that is claimed by their proponents to be an excellent form of communication between product own and sprint development team [18]. Scrum utilises a *product backlog* which is a list of features that are rank ordered by the product owner or client on behalf of whom the features will be developed. The product backlog contains a wish list of features called *product backlog items (PBIs)* generally expressed as *user stories* in the form "As a (role), I want (feature) so that (benefit)". PBIs are meant to have a number of features. They should be independent of each other, negotiable and have business value of some kind. They also must be small, estimable and testable. A redaction of the product backlog is referred to as a *release backlog* comprising PBIs that deliver a subset of the required functionality of the system. Parts of the release backlogs are then allocated to sprint backlogs during the *sprint planning meeting*. The PBIs on a sprint backlog will be developed, tested and shipped by a development team (comprising 4 to 9 people) during fixed-duration iterations called *sprints* (generally 30 days in length). A project management tool used during the execution of the sprint is known as a *burn-down or velocity chart* and this represents the work that remains to be done in completing the PBIs on the sprint backlog. Short *daily scrum meetings* allow the sprint team to quickly report on whether the work is on track and to determine whether the required resources are available. At the end of the sprint, a *sprint review meeting* demonstrates the developed prototype, and a subsequent *sprint retrospective meeting* reviews the process of development and how this can be improved [14]. In terms of Scrum's participants, the sprints are facilitated by a *scrum master* who is responsible for the development process and while having a leadership role they do not have management authority over the team. The *scrum development team* is cross functional and has responsibility for developing the product. A *product owner* is responsible for the vision of the product. While anyone can add to the PBIs, only the product owner is permitted to prioritise

them and is the final arbiter of requirements questions. It is the role of the product owner to be focused on the 'what' rather than the 'how' and this is the justification for the use of user stories, as an accessible straight forward way of representing the needs of those who will use the system rather than the technical interests of those who might build them [18,13].

In this paper we seek to understand how user stories function as communication. In order to do so we first need to identify the kind of communication theory we need and to select an appropriate one to apply to this task. We want to explore what applying an appropriate communication approach tell us about user stories, PBIs, Scrum and systems development in general. This paper is also an attempt to map out some of the implications of thinking about IS development methods as communication. In doing so we hope to better understand what we are doing when applying these methods. By using appropriate communication theory and methods, it may be possible to improve these methods or even create new ones. The structure of the paper is as follows. Section 2 describes how IS Development Methods can be viewed as Communication. We select a linguistic theory called Systemic Functional Linguistics (SFL) to provide basic descriptions of what IS development methods look like from this perspective. In section 3, we apply SFL principles to try to answer the question of what exactly is a user story. In section 4, SFL is applied to identifying and pruning overly complex PBIs referred to as *epics*- the bane of all sprints. By doing so we demonstrate how communication approaches can yield insights into current development practices but is also suggestive of improvements to current development practices. Section 5 provides some conclusions, discussions and proposes further research.

2. IS Development Methods as Communication

2.1 Communication Approaches and Domain Viewpoints

Communication approaches vary to the extent that they can provide complete descriptions of work practices or describe organisational discourse in general. A distinction should be made between a *communication resource*, that describes a facet of language use or activity, or a *communication theory* that consists of a comprehensive range of communication resources and one or more grammars that are used to account for discourse in general. Perhaps the best known communication approach in our discipline is *speech act theory* (SAT) developed by Austin and Searle [3,20]. Speech Acts have been applied in IS [16,10,1] in a variety of specialist and mainstream venues, where they have been put to good use providing new kinds of process descriptions. Indeed [19] demonstrated how speech acts have been used to form executable process descriptions. But speech acts are just one kind of communication resource. Parenthetically, SFL has a system of *speech functions* that removes several known difficulties in applying speech acts; for example the latter's inability to provide grammatical evidence to help in differentiating between competing speech act interpretations.

In exploring the kinds of IS relevant descriptions that could be formed using communication approaches, [5] identified three possible *domain viewpoints* where they could be applied. These can be thought of as a stack. The first viewpoint at the base of the stack is that of *work practices*- the communicative description of either business processes or services. Much of the work of communication-based approaches in IS centres on demonstrating the possibilities of these kinds of representations and in some cases developing systems directly from them. The second viewpoint involves studying the work practices of developers- the *practitioner* viewpoint. The concern of this paper, it is something that is rarely considered in IS communities that utilise communication approaches. The third and final viewpoint at the top of the stack is the *disciplinary* viewpoint concerned with the discursive construction of the discipline; the process orientation in IS and management innovation studies is an example of this viewpoint. We will not consider this viewpoint further here. Communication approaches in the IS discipline differ in the extent to which they can account for generalised organisational discourse, and their ability to do so determines if they can be applied to the practitioner and disciplinary viewpoints at all. Comprehensive communication

theories are likely to handle practitioner and disciplinary viewpoints, while approaches that rely on communication resources are more likely to be limited to the work practice viewpoint.

There are a large variety of comprehensive communication approaches that are potentially applicable to IS. *Formal theories of communication* focus on the structure of individual sentences have proved to be very popular in areas like machine translation of language and computational linguistics in general. But functional communication theories view language as a resource for making-meaning and focus on the uses to which it is applied. From our perspective, the only real choice is to select a *functional communication theory*. Their applied focus is commensurate with the pragmatic and utilitarian goals of systems development discipline itself. The single most significant functional grammar in recent years [8] has been Systemic Functional Linguistics or SFL developed by Halliday and colleagues [12,17,15]. It is the only complete functional communication theory based on semiotics [11]. It has also had a degree of representation within the IS discipline itself [2,4]. We will use SFL in the remainder of this paper.

2.2 Methods as Communication: Basic Descriptions

The communication theory used to theorise ISD methods will determine how methods can be construed. From an SFL perspective, an IS method involves one or more contextually circumscribed completed acts of communication in conventional places or spaces. Completed acts of communication (texts) are situated in an immediate situational context (referred to as *register*) that tells those involved in the communication what is going on (*field*), the social relationships that are taking part in the communication (*tenor*), and how language is being used (*mode*). When we read or listen to a completed act of communication we are able to reconstruct the situational context because there is a systematic relationship between contexts and text. Each of Scrum's stages and their development practices are organised in terms of material settings in which communication occurs and the texts and their associated contexts that are likely to be specific to and characteristic of the specific method and/or methodology. An example of a text that precedes a sprint is the *product backlog*. It is a list in which PBIs are arranged in rank order. After resorting and refinement these PBIs will eventually be implemented as features. The product backlog is a text in which all the sprint team members and the sprint master can add *product backlog items* (PBIs) in user story form for example, but only the product owner gets to prioritise this list. The user stories are also completed acts of communication. In section 3 we turn our attention to describing what kind of texts they are.

The physical space and place in which texts are likely to occur both shape the generic and register characteristics of the text and in turn this textual activity also has an effect on how the space or place is used. The spaces or places are referred to as the *material setting* of the text. Of course most development work is undertaken during meetings, but even here the requirements of some of the development tasks necessitate novel ways of using the space which might appear to be cultural to some extent. Some well-meaning but none-the-less unusual practices have arisen in these materials spaces. One of these practices effectively demonstrates the cultural specificity of developer practices and the need to better understand communication resources. As someone is about to report their activities during a daily scrum meeting, the scrum master throws a ball to a team member who is about to make their report. The ball is used to show the team 'who has the floor'. Only that person is allowed to speak until they are finished or are asked to do so by the scrum master, in which case they throw the ball back to the scrum master. The folk linguistic explanation for this practice is that it puts everyone on the same level politically which is to imply that this practice somehow flattens the unequal social relations of power between members of the team. No amount of ball throwing will reduce social distances or diminish unequal social relations of power within the team. What the ball might show (when and if it works) is the *turn taking* that is occurring in the daily sprint meeting. But anyone could determine the turns taken in a meeting by either being there or reading a transcript of it. We mentioned this kind of practice was cultural; while it probably might work in North America, in Australia tossing a ball at someone might result in it being tossed right back at you- hard! Daily sprint meetings are also often

conducted standing- this helps keeps the meetings short, unless you work in Scandinavia where many knowledge workers have desks that enable them to conduct their work while standing- for well documented health reasons.

We have not yet defined the cultural contexts associated with any IS method from a communication perspective- all texts exhibit high level patterning. For example, when we read and write emails we expect to see stages that identify the expected receiver of the text, the author of the text, the subject and the date. We also expect a message that forms the body of the text. Each of these stages is referred to as the *genre elements* of the text and the order in which they occur, and that characterizes memos from some of the text pattern- is referred to as a *genre*. Collections of the same kind of texts exhibit the same genre. So important is genre to texts (as a kind of text type) that IS artefacts effectively codify them for applications, for example, data entry screens, user interfaces and printed reports are all examples of generic organisation. This has important implications for when considering IS development methods as communication. Not only do methods consist of texts, but these texts can be generalized according to their generic structure. Scrum as a IS development method that has a sprint planning meeting that consists of pre-defined text types like agendas, a release backlog based on a product backlog (all written texts), the meeting itself consists of an extended spoken language text. Eventually this spoken language is transformed into a much simplified written transcript and eventually the minutes of the sprint meeting. Also this stage results in another written text the sprint backlog- a list of written language PBIs. In the next section we apply SFL to unpack some of the myths surrounding the nature of user stories.

3. User Stories considered using SFL

3.1 Better because they are spoken…

One of the often held advantages of user stories is that they shift the focus from writing about requirements to talking about them; [6] claims that "[u]ser stories emphasize verbal communication. Written language is often very imprecise, and there's no guarantee that a customer and developer will interpret a statement in the same way". Here supporters of user stories seem to be privileging speech over writing. The first thing to consider is that speech is not any more or less precise than writing. Rather than considering communication in terms of 'precision' it is more useful to consider it from the semiotic idea of *polysemy*. Both these forms of language support many different interpretations and therefore many different possible meanings.

There is certainly a difference between written and spoken language. In the evolution of language, speech came first, only later with the development of the large scale social organisation and the need to effectively record administrative details, amounts and quantities, did writing develop [11]. Why do functional linguists make a distinction then between written and spoken language? They do so because when we look at the language resources used in both, we see that they are not necessarily the same. Think of the tone resource. We raise our tone at the end of a sentence to indicate a question. But if a speaker is in a very insecure or stressful situation and feeling out of their depth, especially if they are young and/or relatively inexperienced then they may unwittingly indicate their insecurity by raising tones at the end of every utterance. In writing we have to use the question mark '?' to indicate a question but obviously not directly a tone. Parenthetically, if resources are used in speech and writing then they are often deployed differently, see subsection 3.3 below.

In order to negotiate the provision of information and knowledge or the exchange of goods and services, we use language resources that are available only in spoken language (unless of course we transcribe the speech but that is another matter). Following [8], when we talk to each other we use choices that make us initiators of, or responders to, language (the so-called *speech role*). These choices together with the type of *commodity* we are dealing with, whether we are dealing with negotiating goods and services or supplying information, enable us to select an appropriate *speech function*. An initiator will use one of the following speech

functions: an offer, a command, a statement or a question. A responder can accept or reject an offer, comply with or refuse a command, agree with or contradict a statement or answer or disclaim a question. As mentioned earlier, texts can be analysed for speech functions using grammatical rules. Each speech function has an associated grammatical realisation that enables it to be identified in a text with a relatively high degree of reliability. Speech functions co-occur; a question is often followed by an answer, an offer by an acceptance. This is how interlocutors accomplish work with language, by sequencing speech functions between themselves to make meanings. These sequences of speech functions are called *exchange structures*. Finally, patterns of exchange structures are used in routine or conventional spoken language situations. Buying a loaf of bread at a bakery will employ a conventional pattern of exchange structures as well. This pattern is referred to as a *genre*. In this situation, the set of exchange structures involve selecting the items to buy and determining the price, and then paying for those items. Each *genre element* consists of packets of exchange structures. Of course many commercial transactions like buying bread are so thoroughly conventional and so the speech roles, commodities and moves are well known to interactants- in fact they are their own genre called a *service encounter*. Through the experience of a great many similar (and indeed identical) transactions, interactants have experience of every useful and many not so useful approaches to these kinds of situations.

3.2 … but unfortunately they are written.

The folk linguistic account of user stories can be challenged; user stories are not examples of spoken language! But as we have argued previously, these two modes are different. Writing is not speech written down. If you have ever transcribed authentic spoken language situations as we might find it during development sessions, then there is little doubt that user stories are not authentic examples of spoken language. When we speak in social situations we must collaboratively engage with each other. One interactant might start some idea, and the other might complete it- a phenomena known as *latching*. That sort of phenomena can be used to indicate social relations of power. An interactant might simply cut off the discussion by imposing their belief that it is acceptable based on their perception of having a higher social status. Of course, just because one interactant may have a higher social status does not mean they will necessarily interrupt. Culture plays a part in how these behaviours play out in social settings. Importantly, these kinds of phenomena are not found in written language unless we are transcribing spoken language.

Another line of evidence that confirms that user stories could not possibly be instances of spoken language, or even transcribed spoken language, comes from linguistics, see Table 1 Row 1. Written language is *lexically dense*. By this we mean that it is rich in *nominal groups*- a group of words that consist of a noun and information related to it [9]. Nominal groups provide information on experience. In contrast, when an interactant speaks they may correct what they just said, a phenomena called crossing out. They may use lots of ‘ohs’, ‘arrhs’ and so on to include spaces in their talking that enable them to organise their thoughts about what to say next. Spoken language leaves its mark on the resulting grammar; it is *grammatically complex* as a result. But user stories are lexically complex, especially in the classification of roles (certainly complex for real organisations), in the labelling and description of features and the anticipated or expected benefits they produce. The roles, features and benefits are provided by nominal groups and reveals this simple structure is more lexically dense than grammatically complex. Therefore user stories are examples of written language not spoken language; fortunately user stories are no less authentic because of this.

3.3 User stories are clauses not stories

This still leaves the question of what kind of entity are user stories. From an SFL perspective, user stories are a kind of clause. Clauses are the fundamental unit of the lexico-grammatical organisation of texts. The amazing thing about language is that they weave three different kinds of meaning together simultaneously. The first kind of meaning- *interpersonal*

metafunction, describes how a clause is organized to express interpersonal meanings- we have actually described many of these resources in subsection 3.1 when we dealt with interactants negotiating the provision of information and knowledge or the exchange of goods and services. We will not consider these kinds of meaning further. The second kind of meaning in language relates to the how meanings are packaged into language- referred to as the *textual metafunction* considered in more detail in section 4. The third kind of meaning- *ideational metafunction*- describes how reality is packaged into language both in terms of discrete experiences (*experiential meaning*) and as a flow of experience (*logical meaning*).

First we look at how the flow of experience is represented in a user story- this is primarily a structural view that reveals how meanings are refined by expanding upon the meanings of successive clauses. The user story is such a so-called *complex clause*, see Table 1 Row 1. Clauses enable meanings to be packaged into grammar [7] and the process of identifying constituent clauses is called *clause boundary analysis*. Clauses can be *major clauses* that can stand alone or *minor clauses* that require a major clause to make sense but importantly provide additional details. To identify the number of messages that are being represented in a clause complex, we look for *predictors*- the verb (doing, happening, or being) part of a clause. In Table 1, Row 1 the predictors have been underline. There are two distinct messages (clauses) in this user story. The part of the clause complex 'I want to send out an e-brochure to our former clients' is called an *independent clause* (it stands by itself). It is also a *declarative clause*- a statement responsible for providing information. As a clause 'to advertise our new services' is clearly not capable of making sense by itself. It is referred to as a *non-embedded dependent clause* because it is added as an afterthought to the rest of the clause. Importantly, the non-embedded nature of this dependent clause may be suggestive of alternative things 'to advertise' that may form the basis of additional new PBIs. Of course if these existed, they may be of importance to other roles in the organisation. Looking into the logical meaning of a clause is to take a structural perspective on how these clauses are linked together.

Now we can consider the user story in Table 1 Row 1 from the perspective of experiential meaning by identifying what the clause is doing functionally using what is referred to as a *process*. In this case, the user story is a *material process* something is being done (sent out). User stories are typically about doing things but other processes types are possible in clauses, including thinking things (mental), talking about things verbal), behaving in certain ways (behavioural), existing (existential), identifying or describing attributes (relational processes). We assert that user stories must consist of material processes otherwise they are ill-formed.

We can see that the role part of the user story has no predictor and is therefore a *minor clause*. But the wonderful thing about this minor clause is that it contains a *homophoric reference*- a reference to a role in the organisation being adopted by the product owner. The purpose of this reference is to make sure we don't forget which role the 'I' (read product owner) is playing. Recall that clauses construct three different kinds of meaning. From a textual metafunctional perspective, the minor clause also functions as a so-called *marked topical theme*. The term 'marked' simply means atypical or unusual. In most cases *unmarked themes* are typically used, but user stories employ marked ones. While we could have rewritten this user story to remove this marked topical theme, see Table 1, Row 2, the original version does have the advantage of continually reminding us to describe features directed at supporting the human activity system rather than specifying functions in a computer application.

Table 1. [1] An example of a user story- nominal groups italicized and predictors underlined; and [2] a rewritten user story removing the marked topic theme.

1	As a *laboratory manager*, I want to <u>send out</u> an *e-brochure* to our *former clients* to <u>advertise</u> our *new services*.
2	The *laboratory manager* wants to <u>send out</u> an *e-brochure* to our *former clients* to <u>advertise</u> our *new services*.

User stories are not stories either. We encountered genre resources in relation to spoken service encounters in section 3.1 but they also apply to written texts like memos, bulletins, and press releases. Martin and colleagues [17] have surveyed so-called *canonical genres* that are found in many different areas in society. The basic idea is that each activity or discipline can be characterized by those genres that are most applicable to it. A set of so-called canonical *narrative genres* are described in [17] include specific structures for genres that include recounts, anecdotes, exempla and narratives. The narrative structure has three compulsory genre elements called 'complication', 'evaluation', and 'resolution'. We recognise this kind of classic structure in film, television and novels- but it is completely absent from user stories; so from an SFL perspective user stories aren't!

4. Cutting Epics Down to Size: Applying Thematic Progression

A critical aspect of the success of Scrum development projects is having items in the product backlog that are small enough to be implementable within the timeframe of the sprint and yet still provide business value at least from the perspective of the product owner. Each PBI must be examined in order to identify so-called *epics* [6]. Epics are items that are considered to be too complex to handle within a single PBI and that add time and complexity to a Sprint. Identifying epics and successfully decomposing them is entirely dependent on the expertise and experience of the sprint team members and the scrum master. To the best of these authors' knowledge, no theorized practice has been put forward to account for epic identification and decomposition. However, SFL's *theme resources* are capable of identifying epics.

A *theme* is defined as the starting point of a message- what a clause is about. A theme contains what is familiar or known and takes the first position in the clause [12,8,17]. In the case of Table 1, Row 1, the theme is identified by 'I' in the major clause. There is an interesting feature associated with user stories and theme. The 'laboratory manager' of the minor clause is equivalent to the 'I' of the major clause. This repetition is called a *reiterated subject* [7] and is a characteristic of user stories. Interestingly, this feature is associated with rapid speech. This is probably where the myth about user stories as spoken language originates. The remainder of the clause is referred to as the *rheme*. It contains the rest of what the clause is about, in particular the new information being represented about the theme.

Once information is presented in a text it can then be further developed [9]. Information can be picked up as a theme at the beginning of the clause or it can be accumulated as new information in the rheme at the end of the clause. User stories are created out of relevant documents in the organisation or discussions with stakeholders that are eventually represented by the product owner as user stories during sprints. When applied to these kinds of extended texts, an analysis of theme describes how the meanings in a text are thematically developed as the talking or reading continues, while an analysis of rheme shows how new information is distributed and accumulated through the text.

An expectation of the development team is that the user stories lead to PBIs that represent coherent and cohesive system functionality. Written language documents from the workplace, or spoken language interview transcripts with organisational stakeholders, can be analysed for theme to identify unambiguously the actual organisational stakeholders around which user stories should be subsequently created. These texts typically have patterns of *theme reiteration* [7] that when represented as users stories will provide descriptions of related and perhaps even duplicate functionality. Analysing the associated rhemes can be useful in signalling confused information in user stories, typically in the form of duplicate or overlapping information across multiple user stories. All of the user stories associated with a particular stakeholder can be clustered together and rewritten into a single large text to provide effectively a list of all their required system features. This 'constructed epic' can then be split into its constituent clauses using the clause boundary analysis. Each clause can be analysed for its thematic resources. Irrelevant themes can be excluded from the list of potential user story to develop. We contend that a rheme analysis will help to clarify and identify the actual new features and their characteristics- these will be added to the list of user

stories to generate. Rhemes that are duplicates can be merged into one user story; complex rhemes can be factored into simpler ones and each would be tested for distinctiveness.

5. Conclusions, Discussion and Further Research

Almost every communicative characteristic Scrum practitioners attribute to user stories can be challenged. User stories are commonly referred to as instances of spoken language. However, they are manifestly examples of written language and contain almost none of the tell-tale language resources we expect to see in speech- even if it were transcribed. The only exception to this is a single feature (the reiterated subject) that while associated with rapid speech really serves as a reminder to a product owner as to which role they are playing during a sprint. User stories are also not actually stories.

In this paper we have argued that user stories can be better understood if they are viewed from the perspective of an actual theory of communication. Appropriate characteristics of a communication theory to apply to user stories include that they must be comprehensive, complete and functional so that we can better understand what user stories are, how they function and, as a consequence, why they actually work as they do. We selected SFL and identified user stories as complex clauses. Amongst other things, clauses can be interpreted from multiple perspectives simultaneously; as experiences (experiential) and also how themes and information can be woven together to form extended messages (textual). User stories must not consist of too many constituent clauses, and must reduce circumstantial or dependent clauses to assist in making the resulting user stories smaller, estimable and testable. Well-formed user cases will utilise material processes, that is, they must package experience as 'doing'. The structure of user stories includes language features like marked themes that continually reorient the 'story' back to the organisation and the task and therefore preventing a mind-set that views a feature for a human activity system as a function for a computer application. We also suggest that SFL resources be used to identify anomalous distributions of theme that can signal epics- overly complex PBIs that can then be decomposed or factored into usable user stories. This appears to be the first theorised method for dealing with epics in the Scrum literature.

Despite the 'folk-linguistics' that surrounds user stories as communication, they nonetheless work well because of some of their characteristics. These include the use of a homophoric reference back to the human activity system that is of interest during the sprint; the need for relative simplicity in the organisation of clause complexes, and the exclusive use of material processes. Communication theory can therefore describe why user stories actually function to promote communication between product owners and the development team. We have also demonstrated how communication theory, specifically SFL, can theorise and provide methods for tackling important problems in practical Scrum deployment. For example, we demonstrate how theme resources can deal with epics. In so doing, we have demonstrated that appropriate communication theory and methods have the potential to improve mainstream IS development practices.

Future research will be directed at conducting large scale studies that will no doubt uncover other, as yet unexpected, communicative phenomena during IS development. One aspect of development that is not considered in Scrum is how the user stories represent the actual needs of the roles that are being represented in the user stories- in effect the link between the product owner and their representation of the organisational stakeholders wants and needs in user stories. It is also possible to study how PBIs are reinterpreted and how the meanings change (and in some cases slip). While identifying linguistic resources is one thing, creating useful methods to be applied in the field is another direction this research must take before it can have an impact on actual development practices. This paper is just one small step in considering system development methods as a form of communication between stakeholders.

References

1. Ågerfalk, P. J: Information Systems Actability. Understanding Information Technology as a Tool for Business Action and Communication Linkoping Studies in Information Science, Dissertation No 7, Linköping University Press, Sweden, Linköping (2003)
2. Anderson, P. B.: A Theory of Computer Semiotics: Semiotic approaches to the construction and assessment of computer systems Cambridge Series on Human-Computer Interaction. Cambridge University Press, New York (1990)
3. Austin, J. L.: How to Do Things with Words. Oxford Press, Oxford University (1955/1962)
4. Clarke, R. J.: An Information Systems in its Organisational Contexts: A Diachronic Systemic Semiotic Case Study. Unpublished PhD Dissertation, Australia: University of Wollongong (2000)
5. Clarke, R. J.: Communication and Coordination- Fundamental Issues in Business Processes? International Workshop on Communication and Coordination in Business Processes. Sweden, Kiruna. June 22 (2005)
6. Cohn, M. (2009), http://www.mountaingoatsoftware.com/agil/user-stories. Accessed April 30, 2014
7. Eggins, S.: An Introduction to Systemic Functional Linguistics 2nd Edition. Continuum: New York (2004)
8. Eggins, S., Slade, D.: Analysing Casual Conversation. Cassell: London and Washington (1997)
9. Gerot, L., Wignell, P.: *Making Sense of Functional Grammar: An Introductory Workbook* Cammeray, Australia: Antipodean Educational Enterprises (1994)
10. Goldkuhl, G: Information as Action and Communication. In Dahlbom B (ed.) The Infological Equation- Essays in Honor of Borge Langefors, pp. 63-79. Goteborg Un iversity, Goteborg, Sweden (1995)
11. Halliday, M. A. K.: Language as Social Semiotic: The social interpretation of language and meaning. Edward Arnold, London (1978)
12. Halliday, M. A. K.: An Introduction to Functional Grammar. Edward Arnold, London (1985)
13. Highsmith, J.: Agile Software Development Ecosystems. Addison Wesley, Boston: (2002)
14. James, M. (2014), Scrum Reference Card http://scrumreferencecard.com/. Accessed April 30, 2014
15. Kress, G.: Linguistic processes in sociocultural practice. ECS806 Sociocultural aspects of language and education. Deakin University, Victoria Australia (1985)
16. Lyytinen, K.: Implications of Theories of Language for Information Systems. MIS Quarterly. March 61-72 (1985)
17. Martin, J. R.: English Text: System and Structure. John Benjamins, Amsterdam (1992)
18. Pham, A., Pham, P.-V: Scrum in Action: Agile Software Project Management and Development. Course Technology, Australia (2012)
19. Schoop, M.: The Worlds of Negotiation. In: Arkhus, M., Lind, M. (eds.) Proceedings of the 9th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2004), Rutgers University, The State University of New Jersey. NJ, New Brunswick, June 2-3, (2004) www.scils.rutgers.edu/lap04/lap04 Accessed April 30, 2014
20. Searle, J. R.: Speech Acts- An Essay in the Philosophy of Language. Cambridge University Press, Cambridge (1969)