Aram and Neumann *Journal of Internet Services and Applications*
DOI 10.1186/s13174-015-0030-8

Journal of Internet Services and Applications
a SpringerOpen Journal

RESEARCH Open Access

Multilayered analysis of co-development of business information systems

Michael Aram*† and Gustaf Neumann†

Abstract

Business information systems (BIS) comprise technological (e.g. programs), informational (e.g. content) and social artifacts (e.g. collaboration structures). Typically, such systems are constantly and collectively developed (co-developed) further by a variety of individuals within the organization. By recognizing these varying types of actors (concerning their goals, technical expertise and language means) and their predominantly developed artifact type, one can distinguish two types of subsystems: *technical subsystems* wherein the development of the system behavior is conducted by software developers; and *business subsystems* dominated by end-users developing informational artifacts. So far, co-development structures within and between these subsystems are not well understood, especially the aspect that – potentially driven by appropriate measures such as the provision of domain-specific languages – co-development might shift between these subsystems.

This paper presents an approach for characterizing the co-development of real-world BIS with respect to direct participation from different kinds of contributors. This multilayered approach allows us to analyze the co-development with programming languages, domain-specific languages and end-user tools. The approach is suited to assess the direct participation of individuals from different subsystems in the development of evolving BIS. We focus on the intersection of these subsystems, present appropriate metrics and a multilayered analysis scheme. Contributions to artifacts are analyzed using social network analysis to detect structural properties of continuous co-development.

The application to Learn@WU, a real-world BIS, demonstrates how end-user enabling technologies have shifted the co-development effort of the system from a small group of developers to a several orders of magnitude larger group of contributors. We observed an increase of direct participation over time on both informational and executable artifacts, while the number of technical experts was more or less constant.

Our approach may act as a trigger for the application and further development of rigorous instruments for assessing co-development of BIS.

Keywords: Software evolution; Social network analysis; Co-development analysis; End-user development; Domain specific languages; Information systems; Technology enhanced learning

1 Introduction and motivation

As today's organizations are coerced to continuously evolve [1], the information systems that pervade throughout these organizations are ever-changing, too. Consequently, the information technology that supports the organization is usually subject to ongoing (re-) development as well [2]. Consider a large company in the automotive industry as an exemplary organization. Its environment (i.e. the economic situation, jurisdiction, technical innovations, ...) is constantly transforming the spectrum of business requirements, and therefore the organization has to be continuously developed further. The information system, as a socio-technical subsystem of the firm, not only supports but also forms its business processes. Depending on the size of the firm, it may comprise several thousand people, most of whom contribute informational resources and actively use the technology; but typically only a relatively small subset (application and content developers) actively enhances the system's behavior. We assent to the view that a democratization of system development [3] has the potential

*Correspondence: michael@aram.at
†Equal contributors
Institute for Information Systems and New Media, WU – Vienna University of Economics and Business, Welthandelsplatz 1, 1020, Vienna, Austria

to reveal important business benefits, as long as appropriate governance means are provided. We refer to the continuous process of enhancing various aspects of the system (informational, technical, social) by manifold types of organizational actors (ranging from developers to end-users) as *co-development* of business information systems. We strive towards enabling the users themselves to continuously co-develop evolutionary business information systems [4].

One of the fundamental challenges in this research field is to find means to increase the overall degree of participation of the entirety of individuals in the design and development of the system, including the information it processes, the social structures it comprises, and the enabling technology [4]. However, the inherent complexity of real-world business information systems makes it hard for researchers and practitioners to discern the actual co-development structures within a given system instance.

Traditional approaches often apply a singular perspective onto either the technical subsystems (e.g. code bases and bug trackers) or the business subsystems (e.g. wiki co-authorship networks). A comprehensive instrument that facilitates fine-grained understanding of the participation of stakeholders in the evolutionary co-development of complex information systems is still missing. Hence, we address primarily the following research question within this paper: *How can we reveal detailed co-development structures within a business information system?* Accordingly, we present two main contributions of this paper: Firstly, we propose an approach based on a multilayered perspective for the analysis of co-development in business information systems that facilitates to examine the interplay of the business and technical subsystems via a deliberate juxtaposition of the co-development structures among individuals in both subsystems. The goal is to reveal detailed co-development structures within business information systems, that can identify development shifts between the subsystems and that can be used to measure the effectiveness of domain-specific languages for co-development. A key property of the presented approach is that it explicitly identifies layers at the intersection of these subsystems within the information system. Secondly, we conducted an in-depth case study that demonstrates an application of this approach to an actual business information system. The approach and its exemplary application provide a practicable template for researchers and practitioners who aim at evaluating the impact of end-user enabling measures taken in the past.

The remainder of this document is structured as follows: Section 2 touches important related research fields that provide the background of this work. Section 3 reflects on co-development of information systems in general. The core contributions, i.e. the approach and its application, are presented in Sections 4 and 5. Section 6 mentions related work and delimits our study from it. Finally, Sections 7 and 8 discuss our work and conclude the paper.

2 Background and preliminaries

Social network analysis. In general, the understanding of complex networks is an emerging and challenging research field [5]. The defining feature of a social network, according to Wasserman and Faust [6], is the relational information about the actors it comprises. For example, scientific collaboration networks represent the connections between (groups of) scientists based on the papers they have published together [7]. Such sociometric relationships [8] can be revealed through various data collection methods, ranging from prescribed communication lines, and subjective judgements of reputation, to the observation of decision or general interaction processes [9]. For example, Lim et al. [10] have developed a method for analyzing social networks of stakeholders of information systems based on recommendations. For identifying key players in such networks, the concept of centrality [11] plays an important role. As the research field matures, the change over time within social networks gains importance [12].

Software engineering. The discipline of software engineering "is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use." [13] It comprises core knowledge areas such as software construction, design and testing, but is usually distinguished from related disciplines like systems engineering, computer science or computer engineering [14]. Domain-specific software engineering [15] is an important future research direction towards a tighter integration of software systems with their application domain. Domain-specific languages (DSLs) [16] are high-level, tailored languages that are – compared to general purpose languages – easier to understand and use by the people within the respective application domain. End-user software engineering [17] is a form of software engineering conducted not by professional engineers, but by business domain experts who need ad-hoc computational support to fulfill their work tasks.

Information systems. An information system can be seen as a system comprising human beings and/or machines which use and/or produce information [18]. This view, which emphasizes the intertwining of people and technology, is also referred to as the "ensemble view of technology" [19]. The term recognizes the fact that a social network is embedded within every information system.

Software plays a central role in computer-supported information systems, which are the kind of systems that most information systems research efforts typically focus on [18]. Hence, while the information systems field is heavily influenced by (and greatly influences) the software engineering community, it often touches in addition various other related fields such as cognitive science, management science, and systems engineering.

In general, a central purpose of information systems is to facilitate collaboration by acting as communication and coordination systems [20]. The construction-oriented communities within the information systems research field [21] have a tendency towards focusing on the software engineering subarea.

Recently, renowned information systems researchers have emphasized the demand for more comprehensive views of the research object. For example, Lee et al. [22] reflect on the development of the information systems discipline within the last decades and conclude with a call for more focus on systemic, organizational, and informational aspects. Based on this, Lee et al. [23] depart from the traditional socio-technical perspective, and propose to reconceptualize the object of research as a compound artifact that explicitly includes an informational component.

3 Co-development of complex information systems

In this section, we first introduce our perspective on information systems as complex artifacts. The remainder of this section follows the conceptual division of Lee et al. [23] and considers co-development of information systems from an informational, a social, and a technological perspective.

3.1 Information systems as complex artifacts

In his seminal book on the sciences of the artificial [24], Simon describes an artifact as an "interface" between an outer environment and an inner environment. Lee et al. [23] summarize Simon's [24] understanding of artificial things as: "*anything* that is made ('synthesized') by human beings is an artifact". This includes physical artifacts such as hardware devices, and abstract artifacts, the "products of the human mind" [25], such as software. The latter may, or may not, have a more (hardware) or less (software) concrete, physical manifestation.

Following this view, we may perceive an information system (an "implemented instantiation" [26]) as a complex compound artifact of an organization that has at least two important, interdependent outer environments: firstly an outer business environment (containing other organizations, markets, government) that influences the goals and requirements of the business subsystems within the information system; and secondly an outer technical environment that both on the one hand imposes constraints and on the other hand acts as an enabling catalyst for achieving the business goals. The latter contains potentially usable hardware and software components. According to this, the information system operates at the intersection of, and mediates between, these two outer environments, which ultimately induce the majority of developments of the system. The inner environment of the information system artifact can also be perceived to have both more technically oriented subsystems and more business oriented subsystems. Both subsystems should be considered "socio-technical-informational", and interconnected. However, within the technical subsystems the socio-technical aspects – and in the business subsystems the "socio-informational" aspects, respectively – tend to play a more dominant role. The contrived visualization in Fig. 1 (Additional file 1) sketches this perception of complex information system artifacts.

3.2 Information perspective on co-development

At the center of any information system are informational artifacts. Hence, information is, of course, a fundamental concept in the information systems field [27]. The exchange of information is the primordial reason for any

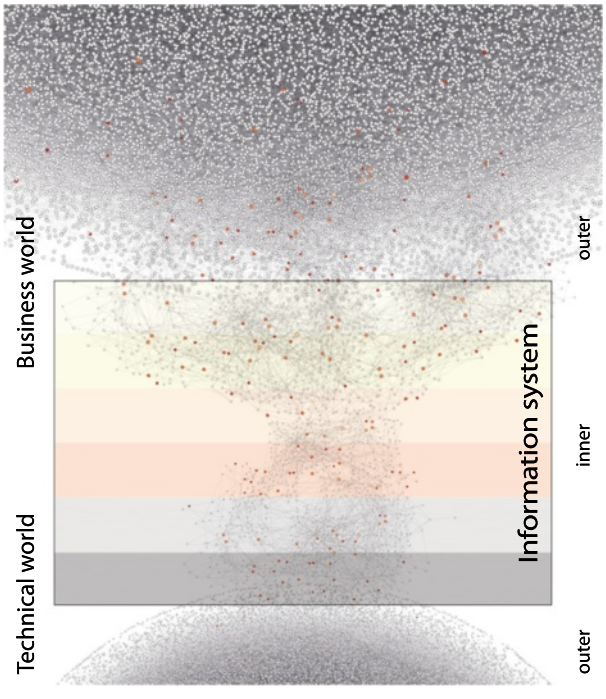

Fig. 1 Business information system. This contrived visualization of a business information system as a complex compound artifact illustrates its interconnected inner subsystems interwoven with its outer environments in the form of a bipartite network comprising artifacts and individuals

business information system [18]. Therefore, designing information and its reception means designing a fundamental artifact within the information system [23]. An attempt to classify all kinds of information perceived and produced by all of the system's stakeholders remains out of the scope of this paper, if at all possible.

Nevertheless, when reflecting the co-development of informational artifacts, one can focus on at least three interdependent aspects: information artifacts as a cause of co-development, information artifacts arising during co-development, and information artifacts as output of co-development. In their seminal article, Germonprez et al. [28] illustrate the concept of *secondary design* by providing an in-depth study of the collaborative construction of an informational artifact.

In this article, we primarily adopt the token-view of information, the predominant view within information systems research: information is seen as "inputs and outputs of processes, in minds, machines, or organizations" [27]. We focus on information artifacts as the product of co-development activities.

3.3 Technology perspective on co-development

When discussing the technology perspective, we apply the same conceptual triad: technology artifacts as enabler of co-development, use of technology artifacts during co-development, and technology as output of co-development. There exists a large amount of research outputs (technological artifacts [29]) aiming at enabling co-development of information systems among contributors of different subsystems.

A prominent role have DSLs, which are specialized languages tailored specifically to an application domain: rather than being made for a general purpose, such languages capture precisely the domain's semantics [30]. DSLs allow to express facts in the idiom and at the level of abstraction of the problem domain. Therefore domain experts may understand, validate, modify and develop the system in a DSL [16]. Examples include high-level workflow definitions [31] expressed in DSL that are easy to understand by non-programmers. There are methods for the rigorous design of DSL [32], which are both a means for facilitating the direct contribution of domain experts to software systems, as well as their collaboration with technical experts [33]. Kelleher and Pausch provide an overview of environments and languages that aim at lowering the barriers to programming for novices [34].

Enterprise wiki systems [35] aim at facilitating contributions to and collaboration on hypertextual information artifacts. By providing end-users with means for creating situational applications, enterprise mashup systems [36] adopt this idea for technological artifacts. Evolutionary information systems [4] aim at combining these properties, hence providing highly introspectable, tailorable technology [37], thus ultimately enabling secondary design at all conceptual layers of the information system.

3.4 Social perspective on co-development

In general, any stakeholder of the information system may contribute to its development. Stakeholders are "the people, groups, or organizations who affect or are affected by a software system" [38], including, but not limited to, software developers, employees, customers, company owners. A traditional mindset considers the technological artifacts of an information system as being designed by software engineers, and subsequently used by end-users.

One of the goals of agile software development methodologies [39] is establishing an environment that facilitates the coordination, collaboration and communication among the members of heterogeneous development teams, consisting of technical and business domain experts. The idea of participatory design [40] refers to integrating non-programmers into the software design process. However, not withstanding this notion of initially engineered (potentially participatory) design, the idea of secondary design treats end-users as "designers in their own right," who are actively engaged in the ongoing design of the information system within the context of use [28]. Following Barki and Hartwick [41], the construct 'user participation' refers to "the activities performed by users during systems development."

Different stakeholders contribute with different intensity and in different forms to the information system. Types of contributions include the establishment of cooperative work relationships, verbal improvement proposals, technical, monetary or ethical requirements documents, software code, et cetera. For example, Lim et al. [42] measure stakeholder involvement by recognizing system usage, system development, financial investment, managerial decision making, constraint imposition, and threatening of system success. In the context of this paper we broaden the definition of user participation of Barki and Hartwick [41] to explicitly include secondary design: we extend end user participation to refer to all activities performed by stakeholders that contribute to the continuous development of an information system. We focus on collaboration structures as the matter of the social artifacts which arise in the course of co-development (most importantly, contributions to technical and informational artifacts) in the context of a business organization. Metrics of social artifacts, such as the number of contributors and the frequency of contributions, show that social structures may influence the technological artifact, e.g. in terms of software quality [43]; in contrast, Bird et al. [44] compared distributed to collocated development in a large software project and found no significant difference in terms of

code quality. Open-source communities can be seen as an example of social structures that are the result, or by-product, of co-development of open-source software [45].

3.5 Challenges

As elaborated above, we position information systems between the poles of the outer business and the outer technical environments (see Fig. 1, Additional file 1). The inner structure of an information system can be understood as being "narrow-waisted" hyperboloids: business domain experts contribute to and collaborate in the business subsystems, and technicians do so in the technical subsystems; at the intersection of these subsystems, the "waist," participation remains low. An information system technology that would support all stakeholder groups equally according to their respective level of expertise would show a broader waist in such a contrived illustration. The forces of change in the information system come from these poles but also from inside the information system, when users of the information system use it to reinvent and to re-engineer their business functions, processes, and organizations. However, within the information system many different partial domains have to be addressed. Many of these have different application and technical aspects, many of these require certain skills and knowledge from both of these poles, leading to an architecture with multiple layers and diverse participation structures. We consider this as a pivotal problem for information systems research: the fundamental challenge, i.e. finding means to sustainably enable participation of as many stakeholder groups as possible in the continuous development of information systems [4], is not only a stimulus for this work, but rather for a whole range of research efforts, including, but not limited to, the design of DSLs [32] and their collective integration [46] towards domain-specific mashup systems [47].

4 An approach to multilayered analysis of continuous co-development of artifacts in business information systems

In general, information systems "are so complex that it is practically impossible to understand them as a whole" [13]. The same is true for assessing the co-development of several aspects of the information system, especially when large groups of individuals (potentially many thousand) contribute to it. To manage this complexity these systems have to be viewed from various angles. In the following, we present a novel, multilayered approach that aims for a better understanding of the participation of individuals in the evolutionary co-development of business information systems. The purpose of the approach is to provide means for characterizing information systems with respect to co-development by different stakeholder groups. This approach helps to assess the degree of direct participation at various layers of individuals from different domains to the development of a constantly evolving information system.

By actually applying this approach to an existing business information system, we demonstrate in Section 5 how the provisioning of end-user technologies has shifted the development efforts within this particular system from a relatively small group of technical system developers to a (by orders of magnitude) larger group of business domain experts. The study reveals the direct participation of the domain experts, and sheds light on the co-development structures of the business and technical domain experts and on the affected layers.

The approach facilitates measuring the *direct participation* of users in the co-development of business information systems. The direct participation is expressed by the number of individuals providing or modifying artifacts without intermediation and by the number of provided and co-developed artifacts, which are categorized in multiple layers ranging from source code over specifications to content. The approach focuses on *direct contributions*, where an individual either provides or modifies an artifact stored in the system. The artifacts are divided roughly into *informational artifacts* (content, data) and *executable artifacts* (programs, executable specifications in domain-specific languages). By modifying executable artifacts, the functionality and behavior of an information system is directly altered.

In Fig. 2 (Additional file 2) two hypothetical business information systems are characterized based on the co-development structure. It visualizes direct contributions to informational artifacts with light grey arrows, while direct contributions to executable artifacts are illustrated with dark grey arrows.

In case A of Fig. 2 (Additional file 2) technical domain experts are solely providing executable artifacts, while business domain experts provide content and data. In this example technical and business domain experts co-develop the conceptual schemata. The implication of this co-development structure is that every modification of the system behavior has to be performed by technical domain experts, who have to interpret the specifications of the domain experts and implement them. The consequence is that the technical developers tend to become a bottleneck especially when the set of specifications changes or grows. In this kind of system, non-technical stakeholders can only *contribute indirectly* to executable artifacts.

In case B of Fig. 2 (Additional file 2), the situation is different, since in this hypothetical information system, DSLs are used. Business domain experts can *contribute directly* to the executable artifacts. The direct manipulation [48] of executable artifacts is not only a means of cost reduction (compared with the indirect manipulation), but

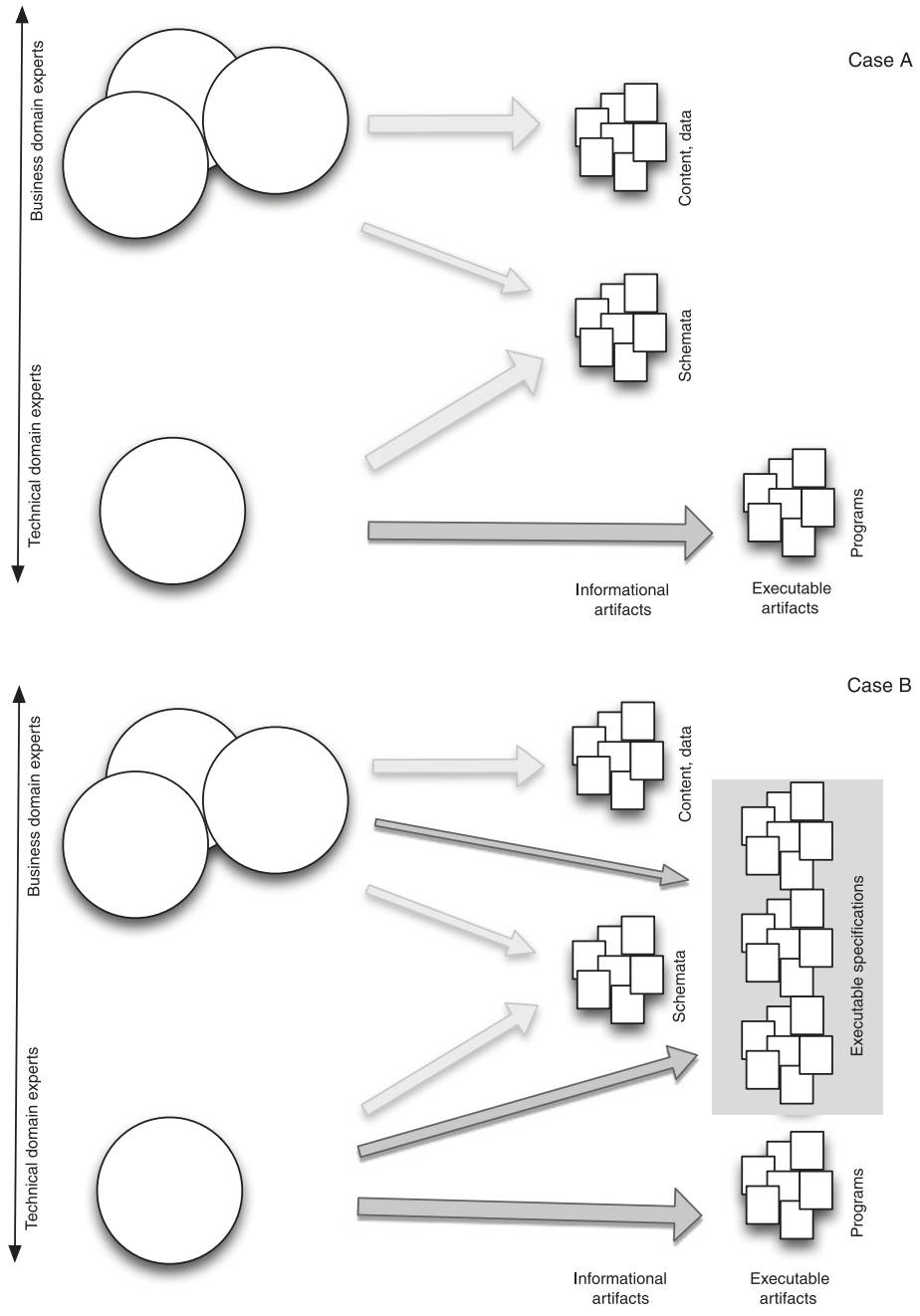

Fig. 2 Different co-development structures in business information systems: business domain experts can contribute to executable artifacts either only indirectly (case **a**) or also directly (case **b**)

finding representations suitable for direct manipulation is an enabler for experimentation and insights. The domain expert is able to experiment with the system in a step-wise manner for developing better solutions. Furthermore, the system in case B enables more business domain experts to contribute effectively to an information system.

The underlying conceptual model is based on a *unified perspective to the informational and executable artifacts* and is in contrast to co-development analysis approaches that concentrate only on the informational [49] or on the software-technical subsystem [50]. As the model of Lee et al. [23] based on social, technological, and informational artifacts is very abstract, our goal is to identify actual instruments to measure and characterize co-development structures. We argue that in the context of co-development a unifying perspective onto executable and informational artifacts is needed to gain a comprehensive picture.

This leads us to a *layering perspective*: important stakeholder groups at the intersection of the business and the technical subsystems act on artifacts that have informational aspects, but which are executable in the information system. A coalescence of informational and executable artifacts can be observed, which is not only noticeable but desirable. Typical examples are executable artifacts specified in DSLs, as addressed in Section 3.3.

Layering. There are several ways to divide a system into layers along the business to technical dimension. Sommerville [13] identifies seven different layers within socio-technical systems. Neumann et al. [4] differentiate broadly between an execution environment, a technical domain environment, and a business domain environment. We argue that in the context of co-development a more detailed layering, particularly within Sommerville's application and business process layers, proves useful. We have identified six layers for the characterization of the co-development structure for information systems development, which are depicted and related to Sommerville's view of socio-technical systems in Fig. 3 (Additional file 3).

Information layer: Content provided and maintained by all kinds of users of the system (including end users).

Configuration layer: Parametrization of the system; allows to choose predefined features, provide specific setup for certain instances, usually via forms.

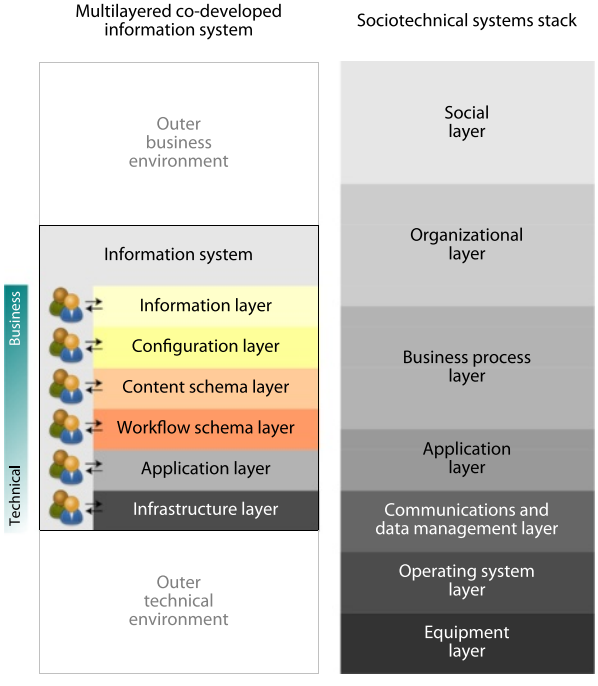

Fig. 3 Layering

Content schema layer: Definition of content (data) schemata to model the application domain, defined in a DSL.

Workflow schema layer: Definitions of workflow schemata to be executed by workflow engines, defined in a DSL.

Application layer: Software components, programmed in generic programming languages for application specific purposes.

Infrastructure layer: Generic (application independent) software components such as database systems, workflow engines, middleware, web servers, operating system.

With respect to participation, artifacts at all of these layers can be designed to enable participation. However, we particularly refer to the two schema layers in Fig. 3 (Additional file 3) as participation-enabling layers, since these are means to enable direct participation of domain experts in tasks that traditionally required technical programming knowledge.

Metrics. In order to obtain a comprehensive, detailed picture of the co-development structure in a business information system, we propose to applying social network analysis techniques at several conceptual layers. We consider the following *essential metrics* to be of interest for most analyses of information system co-development. Contributors and artifacts are the basic types of entities. Contributions and collaborations measure participation in the form of relationships among these entities. These metrics have to be adjusted and supplemented in the course of a concrete research study.

Contributors: We consider a potential contributor as any person who could reasonably contribute to the system. This generally includes all stakeholder groups of the system. A person who actually contributed to the information system is a contributor. The contributor ratio is the amount of actual contributors divided by the number of potential contributors. This figure serves as an indicator of the prevalence of the information system within the organization.

Artifacts: Following the broad understanding of the term artifact introduced earlier, it becomes clear that artifacts exists ubiquitously within an information system. The system itself, as well as its social structures are artificial things. However, social artifacts manifest themselves as the actual collaboration structures that the social network analysis reveals. For the purpose of this metric, one is primarily interested in measurable manifestations of informational and executable artifacts. *Executable specifications* are executable artifacts that explicitly aim at empowering business domain experts to make direct contributions to the system behavior.

Contributions: A contribution is an activity that leads to a modification or an enhancement of an artifact within the information system. In other words, the contribution leads to – thus manifests itself as – the addition, removal, or modification of artifacts within the system. A contribution establishes a a contribution relationship (ConRel) between a contributor and an artifact, the intensity of which can, of course, vary. Laniadio and Tasso [49] incorporate the size and longevity of edits into this measure. Typically, one considers frequency and/or depth [51] as factors to compute the intensity of such a relationship. Particularly when focusing on software quality aspects, the frequency of contributions can be considered to determine a (proportion of) ownership of the artifact [52].

Collaborations: There are several ways to understand the concept collaboration. For example, Briggs et al. [53] identify seven different layers within the concept. Nunamaker et al. [54] distinguish three levels of collaboration, i.e. collective, coordinated, and concerted. Collective collaboration happens when people work individually towards a common goal. More careful coordination of (still individual) work becomes necessary, as soon as dependencies of work increase. When any breaks in work synchronization endanger the common undertaking, concerted collaboration is required.

In general, collaboration can both boost [55] and harm [56] productivity. Hence, while it serves as an appropriate concept for measuring and understanding information system co-development, it should not be misunderstood as an organizational goal per se.

In accordance with our ambitions to define a comprehensive picture of an information system, one may argue in favor of the broadest of these collaboration concepts, i.e. collective collaboration. However, the level(s) of collaboration to be investigated depend on the questions asked in a particular research effort that applies this approach.

Evolution. Within the life time of a business information system its requirements constantly evolve and the participants can change. Hence, we argue that it is necessary to investigate the evolution of co-development within the system over time. We suggest that looking at the history of such a system's co-development patterns provides a very natural way to gain insight. However, depending on the research question, the particular system at hand, and the available data, the time frame selected for investigation may vary.

Scoping. Understanding an information system as an ensemble of social, technical, and informational artifacts implies that its boundaries are neither obvious nor sharp. For example, it is not clear whether or not a customer, or a hardware node, should be considered being part of the system. We have sketched this roughly in Fig. 1 (Additional file 1). As the focus of the framework lies on investigating co-development, we propose to tell "inner" and "outer" artifacts apart as follows. The boundary with respect to people (social artifact) can likely be directly inherited from the business organization. With respect to technical artifacts, those which are continuously developed further by members of the organization are of primary interest. This excludes for example hardware components and off-the-shelf software. The boundaries of the space of informational artifacts are implicitly constrained to those "captured" by the technology and the people involved. Finally, one has to apply a temporal scoping as well, i.e. to decide on the time frame to be investigated.

Data gathering. In general, potential sources for data range from people's answers to survey questions [57], investigating communication in mailing lists [58], over sourcing bug repositories [59], smart phone data [60], software repositories [61] and databases, to log file analysis [62]. Regardless of the methods chosen for a particular study, for reaching an integrated picture of the co-development patterns within the system, researchers should strive for gaining access to data from both the business and the technical subsystems.

5 Application to the educational business information system Learn@WU

The following sections describe the application of our approach through an in-depth exploration of the actual co-development occurring within a real-world educational information system, namely the Learn@WU system [63], a socio-technical system that supports and enables the learning and teaching processes at the Vienna University of Economics and Business [64].

5.1 Study overview

Methodological background. From a broad perspective this study is embedded within our ongoing efforts to investigate analytics-driven, domain-specific information systems [47]. Such a kind of multifaceted research demands a pluralistic understanding [4, 65] of information systems research, combining behavioral and construction-oriented [66, 67] methods.

This particular study represents an in-depth, descriptive, exploratory analysis focusing on a single case [68]. Lee and Baskerville [69] provide an extensive discussion about the generalizability of this kind of research. We study several sources of data and combine them into a common picture. By developing an approach (i.e., a "method" in the terminology of information systems research [29]), this research has construction-oriented [21] elements as well. For example,

Baskerville et al. [70] provide rationale for revealing knowledge that applies to a class of problems through construction of specific solutions.

Scoping of Learn@WU. We have defined the boundaries for our investigation of the Learn@WU system as follows.

Informational: We have investigated all informational artifacts stored within the content repository of the web application framework.

Technological: From the technical perspective, we have drawn the line within the infrastructure layer: the web and database servers are used "as-is," i.e. not co-developed by members of the organization. However, there are packages of the web application framework that both belong to the infrastructure layer and are developed further internally. The latter have been included in the investigation.

Social: The social boundaries in terms of stakeholder groups are determined by the university, and have been restricted to students, teachers, staff.

Temporal: We decided to investigate as much of the systems history as data would allow. As it is the case at many educational institutions, system usage is strongly dependent on the semester terms. For example, during summer holidays system usage decreases significantly. Hence, the university's business years provided appropriate slots for time boxing (note that e.g. the period 2014 should be read as September 1, 2013 until August 31, 2014.)

Layering of Learn@WU. In the following we describe the technological stack of the Learn@WU system according to the layers of a co-developed information system as defined in Section 4 (Fig. 3, Additional file 3). It is built completely on open source components and is running on the GNU Linux operating system. Figure 4 (Additional file 4) assigns its components to the respective layers, which are described in the following.

Information layer: The OpenACS system [71] provides a generic content repository infrastructure, which is used by practically all applications (forums, news, wiki, et cetera) to store content items (informational artifacts).

Configuration layer: End-users can customize and parameterize various system aspects. These tasks are considered as end-user development as well [72, 73]. Although the system allows for fine-grained parametrization and customization of package instances, user-portals etc, an analysis of this layer is outside the scope of this study.

Content schema layer: A flexible enterprise wiki engine allows all stakeholders to define arbitrary content schemata (somewhat similar to Wikipedia's info boxes [74]), which subsequently act as templates for information instances.

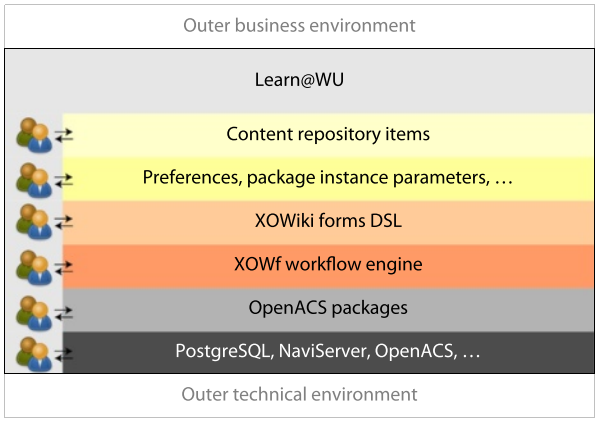

Fig. 4 A layered perspective of the Learn@WU technology stack

Workflow schema layer: The workflow engine enables technical and business stakeholders to define workflows with the help of a DSL. These stakeholders can change several aspects of the system's behavior without having to care about classical programming details.

Application layer: Practically all applications in the Learn@WU system are implemented as so-called packages (installable components) for OpenACS, including the OpenACS base packages or the packages of the learning management system DotLRN [75, 76]. The system uses currently 148 packages (applications and infrastructure), about a third of which have been developed in-house. Ten packages deserve our special attention, as these have been designed with the goal to ease contributions and to foster collaboration within the overall system. Most importantly, these *participation-enabling packages* include a workflow engine [77] and a wiki engine.

Infrastructure layer: PostgreSQL [78] and NaviServer [79] are employed as database management system and web application server, respectively. OpenACS serves as a comprehensive web application framework, and as such, provides an infrastructure for community applications [80]. Demetriou et al. [81] have analyzed the collaboration structure of the OpenACS project.

Metrics for Learn@WU. This section explains the concepts and indicators that we have used in our analysis. In this study we have analyzed more than 5 million contributions from more than 37,000 contributors. All data of this study is based on contributions to digital artifacts, collected from a code repository and a content repository over a period of 10 years. This dataset is the basis of the

social network analysis used to reveal the (changes of the) co-development structure.

Contributors: The most important contributors for Learn@WU are students, teachers, developers. We have calculated the amount of potential contributors for each business year based on the actual amount of students, academic and administration members of the university. The upper limit for the potential users of the system is the number of system accounts, which is determined by the currently enrolled students and employees. Only these have write access to the system and can actually contribute. We have drawn these figures from the yearly reports published by the university [82]. Note, that the numbers of the academic and administration staff had to be approximated, because the university only publishes accurate head-counts for students. The academic and administrative employees are partly declared as 'full-time equivalent' positions, which are less than actual people. Hence, the ratio of employees and students has been approximated as 1:10. However, although an approximation, we are confident that this educated guess is more than accurate enough for our purposes.

Artifacts: One of an educational information system's primordial purposes is delivering learning content to the learner [83]. Content developers continuously develop informational artifacts such as electronic textbooks and questions. However, information instances in such a system are not restricted to learning content in the narrow sense. Learners and teachers shape the overall learning experience by contributing various informational artifacts to the system, e.g. syllabi, calendar entries, forum postings, chat messages, or news. Software developers contribute both informational artifacts, e.g. in the form of wiki pages, and source code. All these artifacts constitute a core asset of the overall information system.

We have mainly considered two manifestation forms of the plethora of artifacts at the various system layers: 'files' at the more technical layers (infrastructure/application), and 'information objects' at the more informational layers (workflow schema/content schema/information). In the context of this study, we refer to workflow schemata and content schemata as the executable specifications within Learn@WU.

Contributions: The source code management system and the web framework's content repository allow us to measure contributions to technical and informational artifacts in a relatively consistent manner. In general, we treat off-the-shelf components as being part of the outer environment. However, in contrast to other artifacts from the outer technical environment, open source software allows for ad-hoc contributions to originally "external" artifacts, transforming them into internally developed artifacts. Hence, we have not counted the first mere *addition* of an externally developed file as a contribution, but its subsequent modifications.

AD, MD, AWD: We used the following degree based metrics for bipartite contribution networks (contributors/artifacts). The average degree of contributions (AD) refers to the average number of contribution relationships of the contributors within the network. The corresponding metric median degree of contributions (MD) is more robust against outliers. The average weighted degree (AWD) incorporates edge weights into the calculation, i.e. the amount of contributions to the same artifact.

Collaborations: In order to gain a comprehensive picture, we have decided to apply the broader notion of "collective collaboration" [54], which includes weaker forms of collaboration, such as asynchronous changes, co-edits, and even competing changes. Hence, each time two distinct persons contribute to the very same artifact within a given time period, we count this as a collaboration. Such collaborations establish a collaboration relationship (ColRel) between the two actors; the artifact becomes a co-artifact. When the same actors work together on another artifact, this establishes a separate collaboration, and strengthens the collaboration relationship. The weighted degree of collaboration (CWD) incorporates these edge weights, i.e. the amount of different artifacts two collaborators have worked on, as a vertex metric. Hence, the collaborator ratio is the amount of actual collaborators divided by the number of potential collaborators, where the latter is the actual contributors in the respective period and context (subsystem/layer).

CAD, CMD, CAWD, CD: For unipartite collaboration networks (collaborating contributors) we employ the following metrics: The average degree of collaboration (CAD) and median degree of collaboration (CMD) refer to the average/mean number of collaboration relationships of the collaborators within the network. The average weighted degree of collaboration (CAWD) incorporates edge weights, i.e. the amount of different artifacts two collaborators have worked on. The density of collaboration (CD) is the ratio of potential collaboration relationships to actual collaboration relationships.

Data of Learn@WU. In our study of the business subsystems, we are concentrating on the direct contributions and collaborations of stakeholders. We analyzed particularly the information, content schema, workflow schema, application and infrastructure layers. The data of the higher layers (information, content schema, workflow schema layer) is stored in a central content repository managed by OpenACS, while the application

and infrastructure layers are managed using a Git [84] repository.

- Information layer (from the content repository)
- Content schema layer (from the content repository)
- Workflow schema layer (from the content repository)
- Application and infrastructure layers (software artifacts from the source code repository)

From the data in the content repository, we were able to analyze 10 business years (2004 – 2014). In this time period over 5 million distinct contributions were recorded by over 37,000 individuals on over 2.2 million artifacts (see Table 1). Such artifacts are for example wiki pages, news entries, syllabi or exercises.

At the application and infrastructure layers we concentrate on the application software artifacts that were modified and extended within the organization, which might or might not have received contributions from outside of the organization. As the software infrastructure is composed of open-source software, there is a potentially wide range of components that serve at the infrastructure layer. Including these outer components in the analysis is beyond the scope of this paper (e.g. the development of the database management system or of the webserver and the programming languages involved). Only the locally maintained components are kept in the source code repository. The analysis of the source code repository comprises the OpenACS packages used, as well as the components developed in-house. The code basis contains 25,544 files (as of 2015), which are programs written mostly in Tcl, JavaScript, SQL and HTML templates as used by OpenACS. Without counting blank and comment lines, these software artifacts amount to 7,833,647 lines of code. From the years 2008 to 2014 we determined 27,000 contributions from 12 contributors to 9,000 software artifacts.

The aggregated, approximated number of potential contributors is about 88,000, while the numbers of potential contributors by year range from about 23,000 to about 30,000 individuals.

Visualization. In this study the data was visualized as two-dimensional graphs using the Gephi [85] software utilizing force-directed algorithms [86, 87]. If not stated otherwise, all graphs follow the same visualization approach: vertices representing participants employ a natural "stellar" metaphor, i.e. those with a stronger influence (weighted degree) are both bigger and brighter (on a gradient from dark red to light yellow). Vertices representing artifacts are only found in bipartite contribution networks and are rendered as equally sized white nodes. Edges are black (slightly transparent) and their thickness relates to contribution/collaboration intensity.

The following sections describe the main part of the analysis. Firstly, we concentrate on the development of contributions to the business and technical subsystems. We can show how the intermediate layers developed over time. Finally, we study how the changes in the artifact structures and layerings are reflected by the collective collaboration structures.

5.2 Contributions to artifacts in the content repository

To study evolution and co-development of the informational artifacts, we need to understand the patterns of informational contributions to the system by the entirety of people involved. The analysis of the corpus of informational artifacts is based on contributions to the system's content repository (see Section 5.1).

5.2.1 Information layer

An initial analysis of contribution-related numbers within the content repository reveals that the information system is used intensively. Within the last ten business years,

Table 1 Contributions to the content repository

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	37427	42.5	2411912	5347213	2252996	64.4	20.0	142.9
2014	17568	70.1	672846	1677392	632325	38.3	17.0	95.5
2013	16548	63.8	538051	1360835	497965	32.5	12.0	82.2
2012	11681	41.3	336557	976296	299211	28.8	3.0	83.6
2011	6836	22.6	216260	409281	208955	31.6	3.0	59.9
2010	7088	24.0	158594	261469	154921	22.4	2.0	36.9
2009	3840	14.4	140115	157708	137789	36.5	14.0	41.1
2008	3460	13.5	119188	135395	118396	34.4	14.0	39.1
2007	4596	18.8	155475	186277	147522	33.8	3.0	40.5
2006	2368	10.0	35426	38325	35144	15.0	3.0	16.2
2005	1054	4.4	16497	18245	16453	15.7	3.0	17.3
2004	176	0.7	23873	25415	23621	135.6	1.0	144.4

over 37,000 contributors (students, teachers, staff) have contributed more than 5 million times to over 2 million informational artifacts (content items). The social network analysis metrics reveal that this translates to an average of about 65 provided/enhanced content items per individual contributor (Table 1, AD).

The evolution of contributions to these artifacts shows growing numbers of contributors, artifacts, and contributions (see Table 1). While in 2004 only 176 people contributed about 25,000 times to about as many informational artifacts, we see 100 times as many individuals as in 2004 contributing to more than 600,000 informational artifacts over 1.6 million times in 2014. These increasing figures are gaining value by the fact that the number of potential contributors is essentially constant. While about one year after the introduction of the system, in the business year 2004, the exploitation of the potential of the information system with respect to its potential users (the contributor ratio) was negligible, it has climbed up to more than two thirds of all potential users contributing in 2014.

5.2.2 Contributions at participation-enabling (schema) layers

The participation-enabling layers (content schema layer, workflow schema layer), are a result of deliberate development measures taken in the past. In order to empower the stakeholders of the information system to contribute more directly to the system behavior, a set of participation-enabling packages (see Section 5.1) had been developed and deployed step-by-step. In 2006 a wiki engine was deployed, which was extended with the functionality for collaborative content schema creation in 2007. Subsequently, in 2008 a workflow engine was deployed, which allowed users to directly adapt the system behavior.

Figure 5 (Additional file 5) shows the evolution of numbers of contributors and contributions to the content repository at the participation-enabling layers compared to the information layer: the white bars act as a reference and show the overall numbers of contributors and their contributions to the content repository; the yellow bars narrow the scope to artifacts that belong to participation-enabling packages, which includes e.g. contributions to purely informational artifacts such as wiki pages; the light orange and dark orange bars narrow it further and show these metrics only with respect to the content schema layer and workflow schema layer, respectively. One can see that after a short lag the provision of each of these participation-enabling packages is followed by successive adoption. The technological support for directly creating and enhancing executable specifications via DSLs acts as a fertile soil for these forms of end-user development.

Content schema layer. We have had a close look at the evolution of the content schema layer: although one business domain expert contributed to this layer from the very beginning, most of the contributors of the first two years were either part of the software development team, or technical experts with close relationships to the team. However, since 2009 the vast majority of contributors actually comes from the business subsystems. During the last eight years, in sum 208 individuals have contributed nearly 4,000 times to half as many content

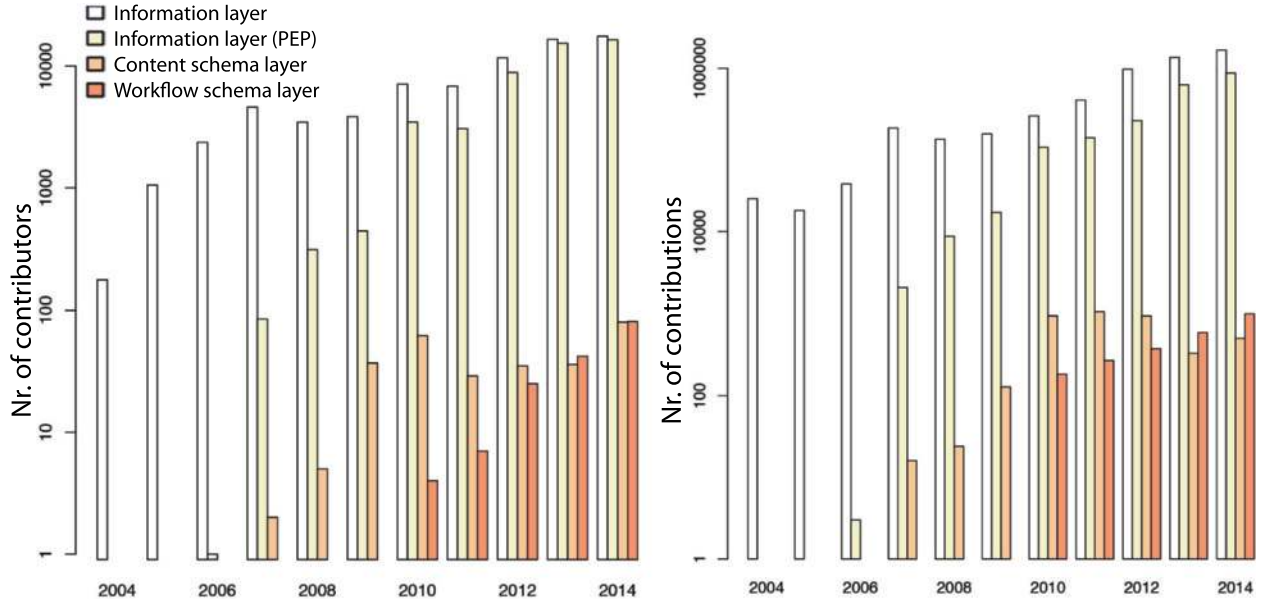

Fig. 5 Evolution of contributions at participation-enabling layers. x-axes: business years • y-axes: logarithmic scale of contributors/contributions

Table 2 Contributions to the content schema layer

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	208	0.2	2187	3981	2003	10.5	2.0	19.1
2014	80	0.3	346	503	268	4.3	3.0	6.3
2013	36	0.1	187	330	184	5.2	2.0	9.2
2012	35	0.1	313	944	270	8.9	2.0	27.0
2011	29	0.1	938	1071	923	32.3	3.0	36.9
2010	62	0.2	328	948	296	5.3	1.5	15.3
2009	37	0.1	81	128	79	2.2	2.0	3.5
2008	5	0.0	9	24	9	1.8	1.0	4.8
2007	2	0.0	2	16	2	1.0	1.0	8.0

schemata, by using a DSL. The social network analysis of the content schema layer reveals that it is populated by a majority of people who contributed to only few content schemata (median degree of contributions of 2; see Table 2). Only a small number of individuals have a broader influence. At this layer, the number of contributors (Fig. 5, Additional file 5) and the contributor ratio peak in 2014. However, other indicators, such as the number of created/enhanced content schemata (artifacts), contributions, and the degree-based metrics, seem to stabilize, or even had peaks in earlier years.

Workflow schema layer. The evolution at the workflow schema layer shows an upward trend. Similar to the content schema layer, a thorough investigation of the workflow schema layer shows that the early two years are dominated by software developers (2010 completely, in 2011 there was one contributor from the business subsystems). Already in 2012, however, we see five times as many business domain experts contributing than software developers; in 2014 the number of software developers at this layer is negligible. Overall, the workflow schema layer has about ten times as many contributors as there are software developers (see Table 3). These domain experts have conducted this form of end-user programming [88] about 2,500 times in the context of 1,500 workflow schemata during the last five years. However, these stakeholders typically contribute to only few different artifacts (overall a median degree of contributions of five), slightly more but similar to the situation the content schema layer. Nevertheless, at the workflow schema layer the number of contributors, contributions, and artifacts have been constantly growing during the last five years. Average and mean of contributions show less increase over time. While the tendency of increasing overall adoption at the workflow schema layer looks promising, the stagnating contribution-related metrics per individual user hint at potential for improvement measures.

To sum up, although these layers are by orders of magnitude less populated than the information layer, both of them are – as we will see later – more crowded than the application layer. The growing contributor ratios at the two schema layers during the last four years show that there is a tendency of increasing adoption. However, as the relatively low median degrees of contributions reveal (see Tables 2 and 3), contributors at these layers spread their contributions to much less different artifacts compared to the other layers.

5.3 Contributions to software artifacts

We continue the analysis top-down the stack of layers by analyzing contributions manifested in the source code repository. While executable artifacts exist in the form of executable specifications (content and workflow schemata) in the content repository, too, the majority of executable artifacts in the analyzed system are classical software artifacts (programs written in Tcl, Javascript, SQL and HTML templates as used by OpenACS).

Table 3 Contributions to the workflow schema layer

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	113	0.1	1526	2445	1495	13.5	5.0	21.6
2014	81	0.3	865	1006	856	10.7	4.0	12.4
2013	42	0.2	374	590	366	8.9	5.0	14.0
2012	25	0.1	226	376	220	9.0	4.0	15.0
2011	7	0.0	46	270	41	6.6	3.0	38.6
2010	4	0.0	20	183	17	5.0	3.5	45.8

Table 4 Contributions to the application and infrastructure layers

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	12	100.0	11187	27825	9062	932.2	171.5	2318.8
2014	7	100.0	925	2306	666	132.1	127.0	329.4
2013	6	100.0	572	1227	469	95.3	100.0	204.5
2012	6	100.0	582	1432	466	97.0	104.5	238.7
2011	7	100.0	866	1791	664	123.7	165.0	255.9
2010	9	100.0	1192	2080	805	132.4	80.0	231.1
2009	7	100.0	8577	12660	8240	1225.3	65.0	1808.6
2008	6	100.0	5556	5831	5375	926.0	66.5	971.8

5.3.1 Application and infrastructure layers

The individuals contributing to these software artifacts form the team of the software developers. The contributions of interest manifest themselves primarily as *modifications* to source code files. Over the last seven years, 12 contributors (software developers) have contributed about 27,000 times to about 9,000 software artifacts (Table 4), these are 28 % of the 25,544 artifacts available in 2015. Since 2008 practically all internally developed software artifacts are managed using a Git [84] source code repository, which receives both updates from internal developers and from the open source community. In fact, this covers all available source code management data at the application and infrastructure layers. Earlier data is not available for all components. The repository served as the source for observing contributions and co-development at these layers.

As an initial overview, we performed a social network analysis of the aggregated contributions leading to a bipartite graph containing contributors and artifacts (files) as vertices (Fig. 6, Additional file 6). The graph visualizes contributions of the software developers of the

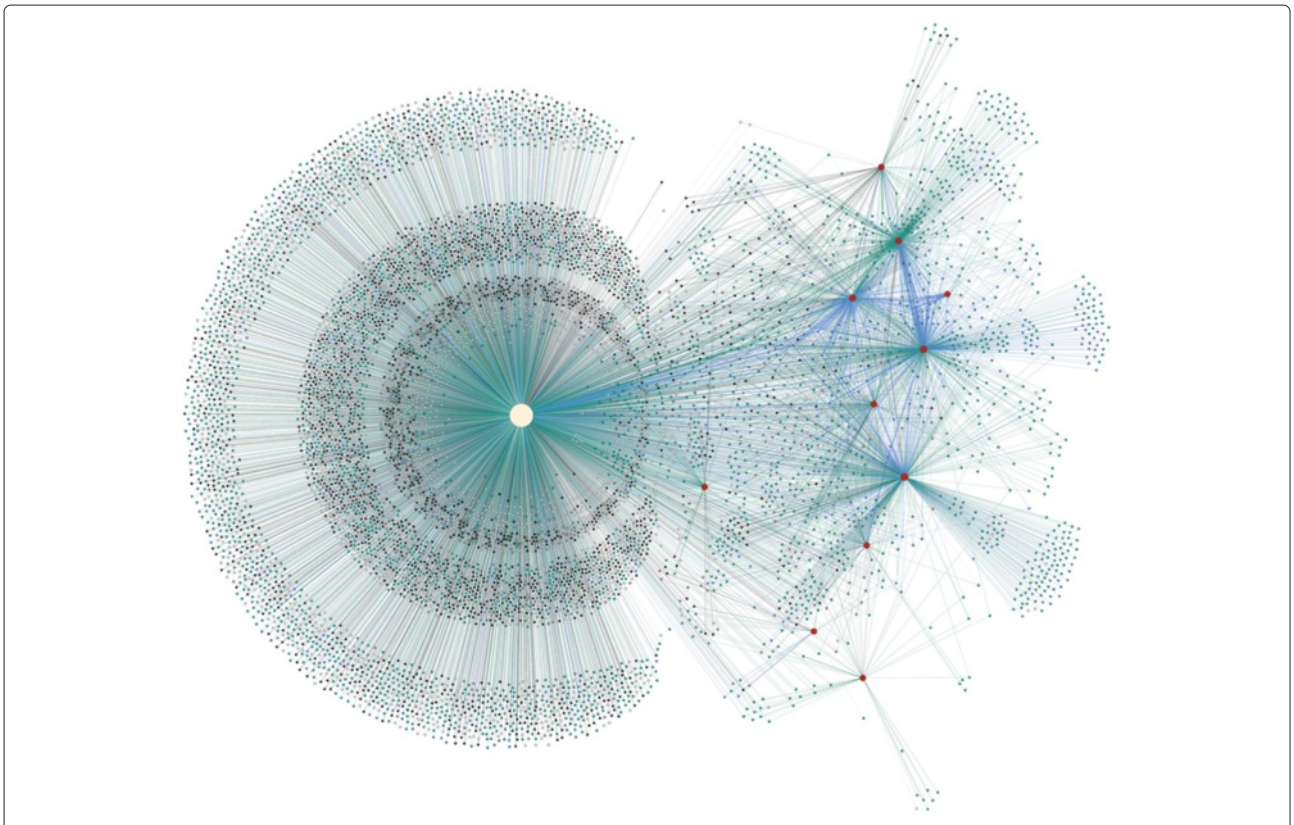

Fig. 6 Contributions in the technical subsystems at the application and infrastructure layers. Vertices [1]: 12 software developers – Vertices [2]: 9,062 artifacts (mostly source code) ● Edges: contribution relationship ● Rating: size of vertices [1] is related to their out-degree. Edge weights are related to contributions. ● Time frame: 2008–2014 ● Layout algorithm: ForceAtlas2 [86]

Table 5 Contributors sorted by overall contributions

Contributor	Contributions	CWD
C1	8504	1826
C2	741	1260
C3	512	981
C4	429	877
C5	334	634
C6	198	435
C7	145	271
C8	105	242
C9	95	227
C10	62	122
C11	33	52
C12	29	103

Learn@WU team to software artifacts at the application and infrastructure layers since 2008; we differentiate graphically between artifacts that were developed mostly in-house (colored) or mostly externally (gray). The size of the contributor vertices is determined by their weighted out-degree, i.e. for a developer by the number of modified artifacts, including modification intensity. While in the overall contribution graph only about a third of nodes are internal artifacts, this number rises to over two thirds when only considering co-artifacts. The graph shows a developer who contributed to a large variety of in-house developed and externally developed artifacts, according to his role in the development team. Most of the other 11 developers focused on in-house developed learning applications (green) or in-house developed infrastructure components (blue). As expected, the collaborative development has a tendency to happen around internal co-artifacts. Table 5 shows that the number of contributions, as well as their respective weighted degree of collaborations, varies greatly among the software developers in the team. While certain developers contribute to a larger number of packages, some other developers have specialized on certain packages, leading to lower weighted degree of collaborations. We see on average nearly 1,000 artifacts enhanced per individual contributor (Table 4, AD).

The evolution of contributions to software artifacts is summarized in Table 4. Compared to the contributions in the content repository, the number of contributions to software artifacts is relatively stable. Within the last seven years, the number of contributors and the median degree of contributions of the software developers has remained in the same order of magnitude.

The visualization in Fig. 7 (Additional file 7) shows, that the variance of contribution spread among the software developers has flattened over the years. It is interesting to see that the number of contributions and affected artifacts (mostly program files) was higher in 2008 and 2009 than in the last two years. There was one developer in these early

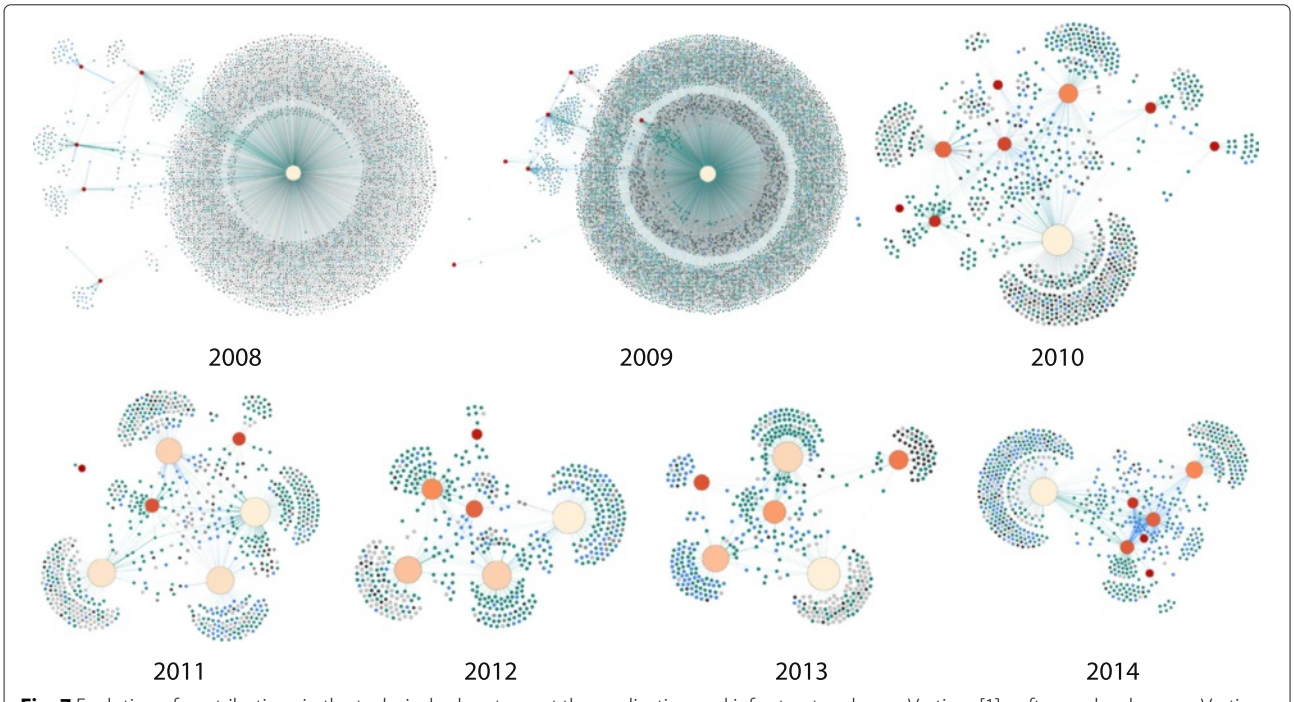

Fig. 7 Evolution of contributions in the technical subsystems at the application and infrastructure layers. Vertices [1]: software developers – Vertices [2]: artifacts (mostly source code) ● Edges: contribution relationship ● Rating: size of vertices [1] is related to their out-degree. Edge weights are related to contributions. ● Time frame: 2008–2014 ● Layout algorithm: ForceAtlas2 [86]

years, who affected large parts of the system practically solely. This suggests more initial development in the earlier years, many relatively stable artifacts. The system contains about 14,000 program files, of which 9,062 received contributions between 2008 and 2014. Only 3 % to 5 % of the total number of artifacts are modified per year. Such contribution patterns which affect a broad range of artifacts with little co-development by others suggest maintenance work (e.g. refactorings) and strongly affect the average degree of contributions. In the subsequent years these differences become much less significant. The last four years of technical contributions provide a much more homogeneous picture with more or less even contributions from every developer (Fig. 7 (Additional file 7), Table 4).

5.4 Collaboration on artifacts in the content repository

In accordance with the analyses of the corpora of informational and software artifacts above, we complete the analysis by studying social artifacts arising during co-development of the system. We focus on social networks that can be observed when analyzing co-development of artifacts. The evolution of the actual social co-development structures manifest themselves as patterns of collective collaboration among the people over time. In order to obtain a vertically integrated view of collective collaboration, we have studied the collaboration in accordance with the identified layering scheme. Figure 8 (Additional file 8) shows the stacked collaboration graphs of the last business year.

The "planned" organizational structures with respect to this educational business information system have been relatively stable over the last years. Only a small group of software developers (currently 7; a total of 12) is in charge of developing the technical subsystems. The university currently employs about 20 so-called eDevelopers and eAssistants to develop learning content based on the instruments provided by the software developers, to support teachers with respect to the technology, and to interact with students in forums or wikis. Finally, teachers and administration use the technology to communicate and collaborate within the system. The staff of the university totals to about 1,000 full-time equivalent employees, the number of students is constantly in the range between 21,000 and 28,000. Practically all these people use the system on a regular basis.

5.4.1 Information layer

At first sight the structures of collective collaboration within Learn@WU are, to a certain degree, as one would expect: analogous to the salient contrast between the corpora of informational and software artifacts, collective collaboration in the business subsystems is overwhelming. The co-development space of the business stakeholders

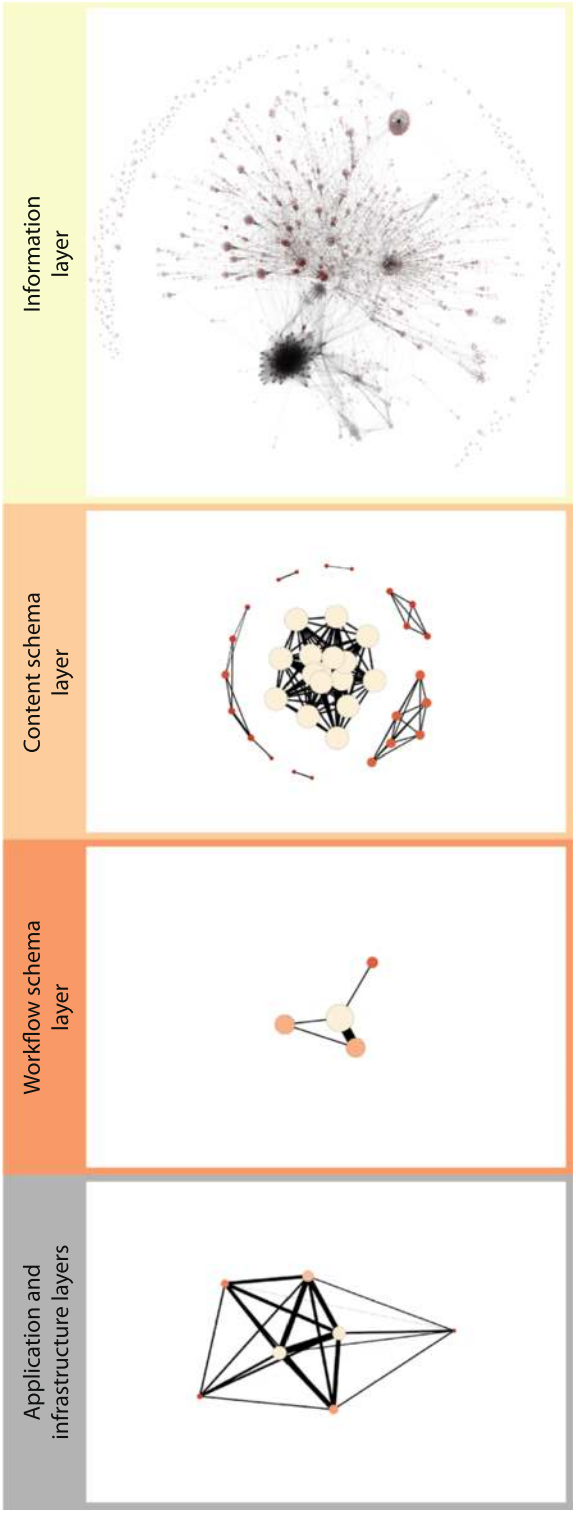

Fig. 8 A multilayered perspective of collaboration on and within participation-enabling packages in 2014. Vertices: collaborators • Edges: collaboration relationships • Rating: size and color saturation of vertices are related to weighted degree. Edge weights are related to collaboration intensity. • Time frame: 2014 • Layout algorithm: ForceAtlas2 [86]

within the information system is by orders of magnitude bigger than the co-development space of technical stakeholders (Fig. 9, Additional file 9). To gain a deeper understanding of the collaboration structures we analyze in the following sections how the collective collaboration has developed over the years and apply the layering scheme presented above.

Over 37,000 individuals contributed to more than 2 million artifacts, and about 41 % of these actually collaborated (Tables 1 and 6). In such a situation, collaboration is unsurprisingly heterogeneous: the bipartite graph (business subsystems in Fig. 9 (Additional file 9)) shows a handful of larger groups collaborating very strongly, but there are a wide range of groups with comparatively low collaboration as well. Over the last ten years, 15,000 collaborators have worked together on nearly 130,000 co-artifacts. While the overall collective collaboration among business stakeholders (co-developments of content repository

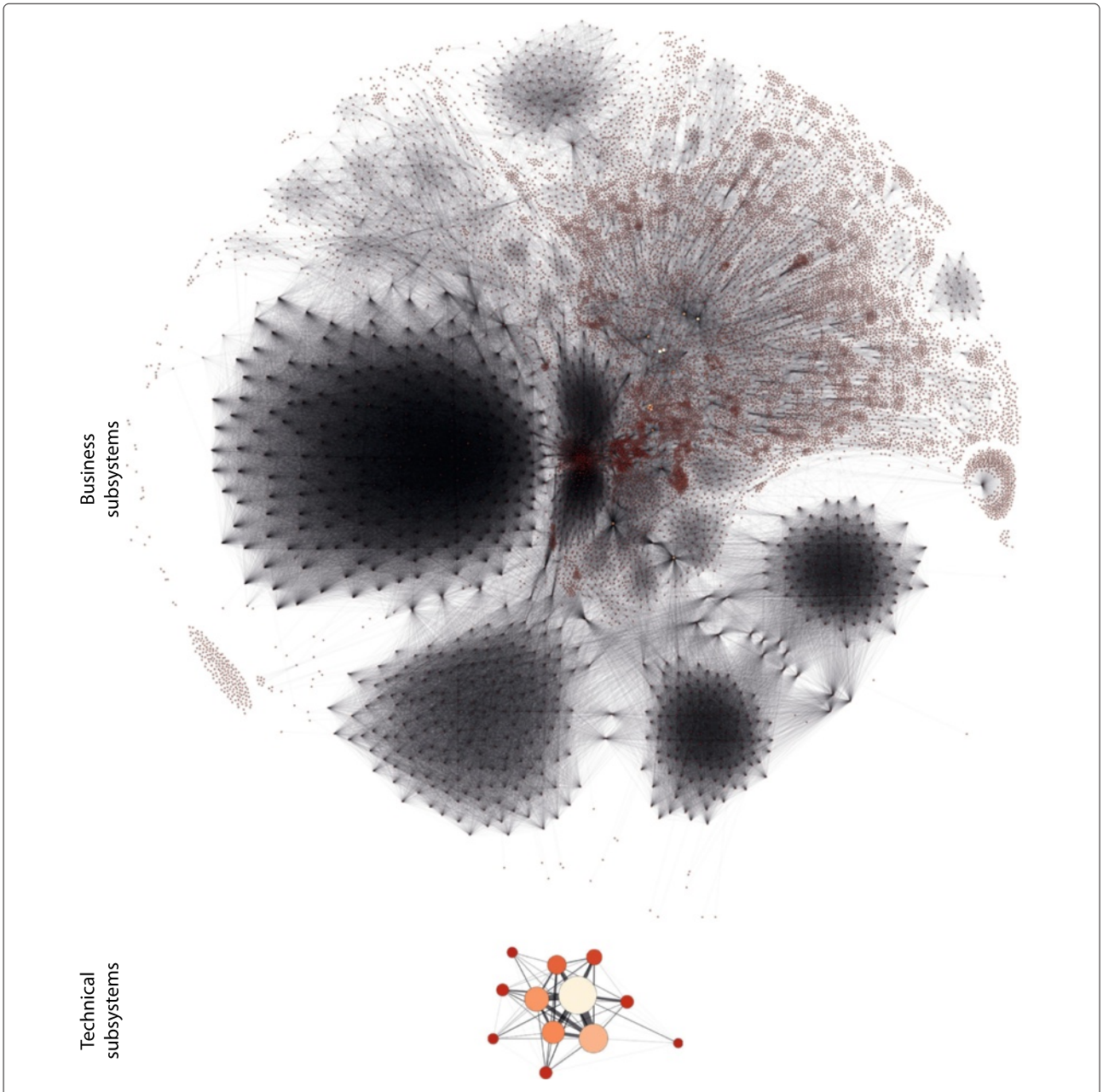

Fig. 9 Collective collaboration within Learn@WU, among business and technical stakeholders, respectively. Vertices: 15,180 collaborators in the business subsystems, 12 collaborators in the technical subsystems • Edges: collaboration relationships • Rating: size and color saturation of vertices are related to weighted degree. Edge weights are related to collaboration intensity. • Time frame: 2004–2014 (business subsystems), 2008–2014 (technical subsystems) • Layout algorithm: ForceAtlas2 [86]

Table 6 Collaboration on artifacts in the content repository

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	40.6	15180	140792	371048	127328	18.5	2.0	48.9	0.001
2014	41.2	7234	16566	53509	35785	4.6	1.0	14.8	0.001
2013	44.4	7346	52630	134178	34041	14.3	1.0	36.5	0.002
2012	33.1	3865	28843	75168	29270	14.9	1.0	38.9	0.004
2011	17.8	1215	13501	26097	4203	22.2	6.0	43.0	0.018
2010	9.4	664	3452	9029	2198	10.4	3.0	27.2	0.016
2009	13.3	510	3149	7381	1120	12.3	5.0	28.9	0.024
2008	9.0	313	720	1283	643	4.6	2.0	8.2	0.015
2007	2.7	126	130	10159	6467	2.1	1.0	161.3	0.017
2006	1.9	46	32	285	279	1.4	1.0	12.4	0.031
2005	3.2	34	21	44	44	1.2	1.0	2.6	0.037
2004	13.6	24	24	255	249	2.0	1.0	21.2	0.087

items) is much higher, from the perspective of an individual collaborator, it is weaker than at the application and infrastructure layers (co-developments of software artifacts): the social network analysis reveals that collaborators typically work with only two other persons (median degree of collaboration of 2.0). The graph in Fig. 9 (Additional file 9) visualizes the aggregate collective collaboration data of the business subsystems, and is therefore by orders of magnitude sparser (CD of 0.001) than the corresponding graph of the technical subsystems.

In the business subsystems, on the contrary, we have 300 times as many collaborators in 2014 compared to 2004. Collaborations and co-artifacts have risen by two orders of magnitude within the same time frame. However, from the perspective of each individual collaborator, the intensity of collaboration does not; as the social network analysis reveals, the degree-based measures have remained in the same order of magnitude since 2008, and have even decreased steadily since 2011. This is reflected in Fig. 10 (Additional file 10), where we see a much more homogeneous structure with respect to co-development in 2014, compared to the years before.

Many clearly recognizable co-development clusters of business stakeholders faded in 2014 towards a more uniform picture. This suggests performing a deeper causal investigation, but is outside the scope of this descriptive study. Nevertheless, the number of collaborators among actual contributors (collaborator ratio) shows a clear tendency to increase: while during the first seven years it was (often markedly) below 15 %, during the recent three years over a third of contributors have collaborated. Hence, despite stagnating figures regarding group size, we can say that the overall collective collaboration within the information system, due to the evolution in the business subsystems (Fig. 10 at the top, Additional file 10), is practically "exploding."

5.4.2 Collaboration at participation-enabling (schema) layers

In contrast to the evolution of contributions at the two schema layers (Section 5.2.2), the evolution of the collaboration structures at these intermediate layers shows broader variance.

Content schema layer. At the content schema layer 88 individuals (mostly business domain experts) have collaborated about 800 times on 80 content schemata since 2009; this is summarized in Table 7. There is no collaboration before 2009, only after adoption of business domain experts people started to collaborate. In the two years with very low collaboration (2009 and 2013), one software developer collaborated with business domain experts on content schemata, most likely in the course of technical support. In 2011, there were four software developers and three business domain experts involved in the collaborative work on content schemata. In 2012 five software developers and nine business domain experts populate this layer.

At this layer, the number of collaborators varies greatly from year to year, which suggests occasional project-based co-development of schemata for learning content. The application of social network analysis techniques, and particularly the visualization (see Fig. 8, Additional file 8), underpins this assumption by clearly showing a large tight cluster of strongly collaborating individuals, surrounded by three smaller teams and three pairs.

Workflow schema layer. At the workflow schema layer (Table 8), which came into being in 2010, we see 43 collaborations from 13 collaborators (domain experts) on 23 workflow schemata. As there were only software developers contributing to this layer in 2010, collaboration happened only among software developers, too. The sole contributor from the business subsystems (see

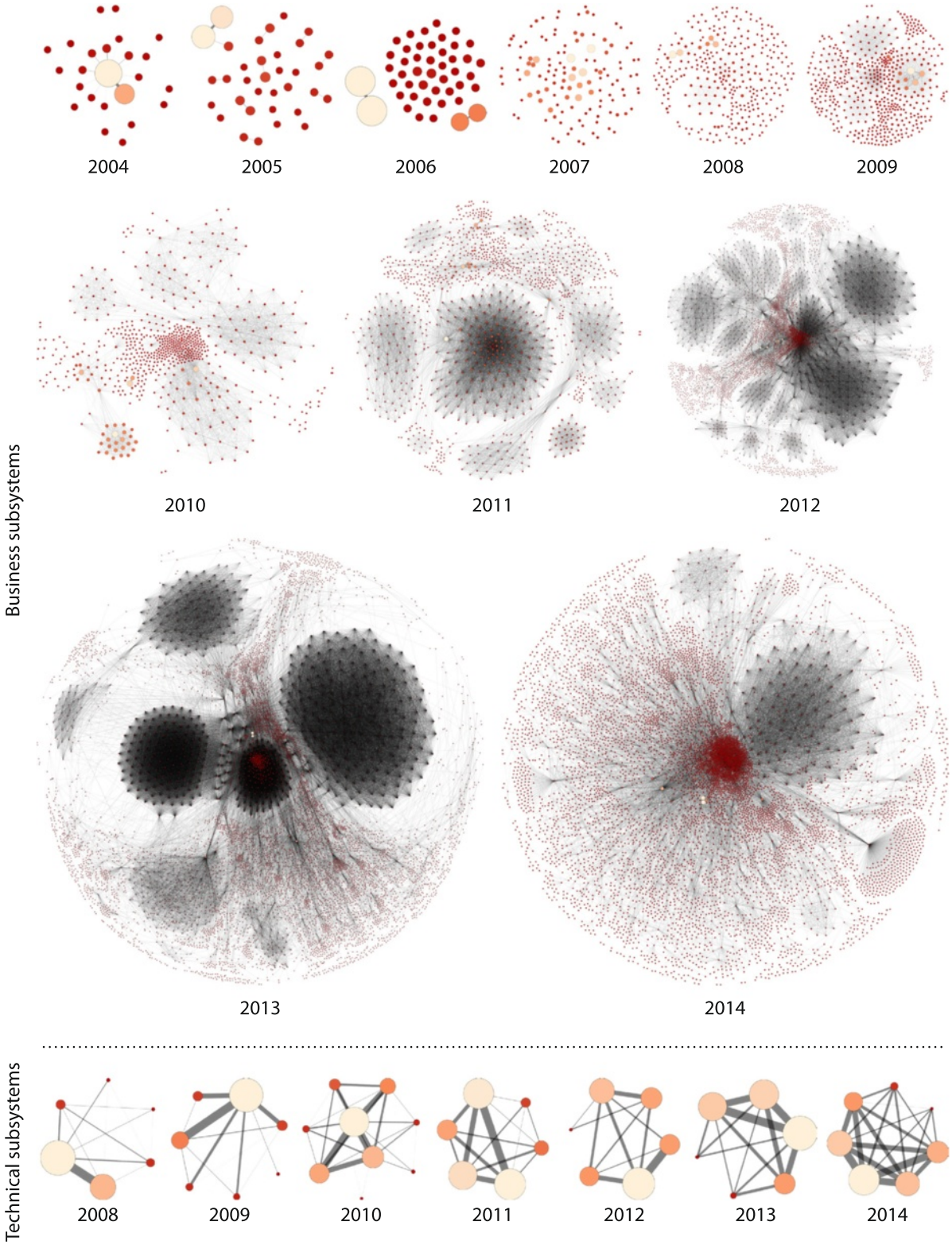

Fig. 10 Evolution of collaboration in Learn@WU, among business and technical stakeholders, respectively. The figure aggregates unipartite graphs that show the collaboration relations (edges) between collaborating stakeholders (vertices) in the Learn@WU system. Size and color of both vertices and edges relate to weighted degree/weight within each graph. Layout algorithms: ForceAtlas2 [86] (business subsystems) and Fruchterman-Reingold [87] (technical subsystems)

Table 7 Collaboration at the content schema layer

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	42.3	88	476	817	80	10.8	7.5	18.6	0.124
2014	47.5	38	152	422	21	8.0	5.0	22.2	0.216
2013	8.3	3	2	3	3	1.3	1.0	2.0	0.667
2012	40.0	14	22	58	32	3.1	2.0	8.3	0.242
2011	24.1	7	10	19	11	2.9	3.0	5.4	0.476
2010	59.7	37	285	286	9	15.4	23.0	15.5	0.428
2009	8.1	3	3	3	1	2.0	2.0	2.0	1.000

Section 5.2.2) in 2011 collaborated with three software developers. In the subsequent business years, each year a varying number of software developers supported two (changing) business domain experts in creating workflow schemata.

As Fig. 11 (Additional file 11) visualizes, collaborations at both schema layers peak in 2014, but there is no clear upward trend in the years before. Collaborations have continuously increased, as Fig. 11 (Additional file 11) shows, but the number of collaborators remained relatively low. As the number of contributors has steadily increased, the collaborator ratio has dropped from 100.0 % in 2010 to 4.9 % in 2014.

This thorough investigation of the co-development structures clearly shows the "bridging" characteristics of these intermediate participation-enabling layers, which facilitate co-development of executable specifications across subsystems.

Finally, it should be noted that both collaborators and collaborations within participation-enabling packages (e.g. wiki page co-edits; yellow bars in Fig. 11 (Additional file 11)) rapidly caught up with the overall collaboration numbers (white bars in Fig. 11 (Additional file 11)). The very same pattern appears with respect to the evolution of contributors and contributions (Fig. 5, Additional file 5). This means, that the users' participation in co-development of the information system successively shifts toward the participation-enabling packages. Although this is a result of deliberate co-development enabling measures, it should not be taken for granted.

5.5 Collaboration on software artifacts

5.5.1 Application and infrastructure layers

An initial analysis of collaboration on software artifacts shows that the collaboration among software developers is very strong (at the bottom of Fig. 9 (Additional file 9)). Every technical contributor to the system is actually a collaborator. Although the aggregate collaboration graph is not complete (CD of 0.894), each developer on average collaborated with 9.8 other developers.

Over the years, this amounts to 3,500 collaborations on more than 1,000 co-artifacts. By ranking the developers according to their weighted degree of collaborations, i.e. incorporating the amount of different co-artifacts they have collaborated on, a dominant collaborator can still be identified. Nevertheless, overall the application and infrastructure layers provide a relatively homogeneous picture of collaboration structures; the graph is close to a complete mesh. With respect to betweenness centrality, closeness centrality and eccentricity (centrality correlates with an actor's coordinative influence [89]), the six most actively collaborating developers share the same values, whereas developers engaged for a short time have a substantially lower degree.

Over the years, the numbers of collaborators, collaboration relationships and co-artifacts vary, but have the same order of magnitude (Table 9). Practically all developers collaborated constantly with each other (with the exception of 2011, when some personnel fluctuation occurred). The number of co-artifacts is rather decreasing, which might be a consequence of a progressing specialization of the stable number of developers.

Table 8 Collaboration at the workflow schema layer

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	11.5	13	23	43	23	3.5	4.0	6.6	0.295
2014	4.9	4	4	10	8	2.0	2.0	5.0	0.667
2013	7.1	3	3	10	6	2.0	2.0	6.7	1.000
2012	24.0	6	6	7	5	2.0	2.0	2.3	0.400
2011	57.1	4	6	8	3	3.0	3.0	4.0	1.000
2010	100.0	4	2	3	3	1.0	1.0	1.5	0.333

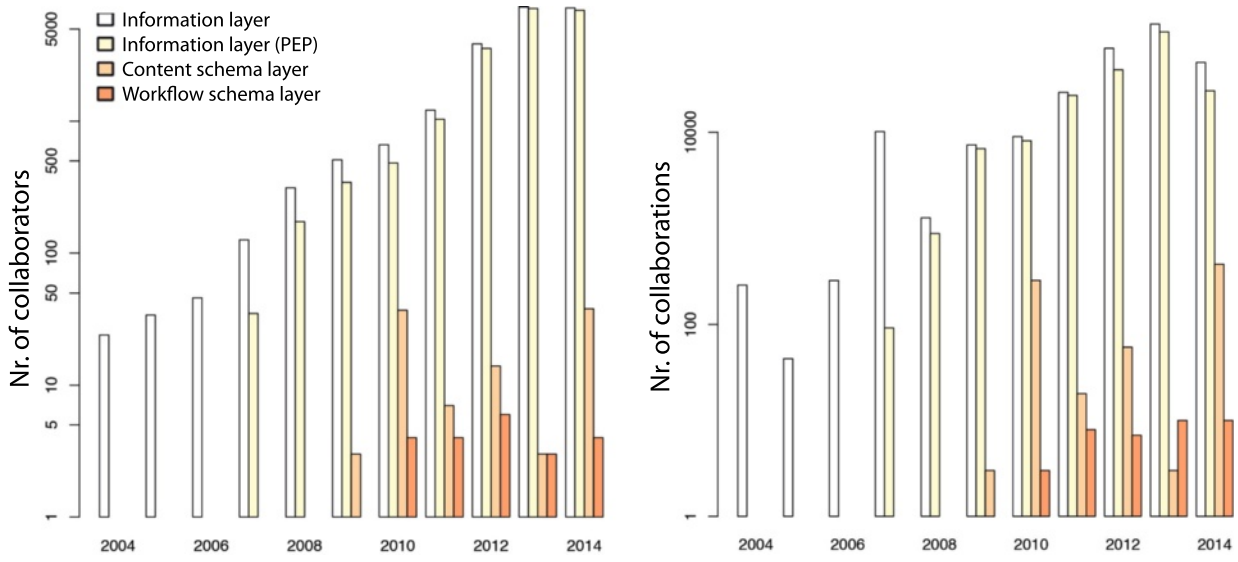

Fig. 11 Evolution of collaborations at participation-enabling layers. x-axes: business years • y-axes: logarithmic scale of collaborators/collaborations

5.6 Contributions and co-development analysis across layers

Finally we look at the differences between the layers with respect to the contribution and collaboration structures. In the global perspective, we could contrast the collaboration from the informational and executable artifacts from the content repository with those from the source-code repository. Looking at all artifacts is not very useful since not all data is technically suited for collaborations. So we concentrate here on the informational layer of the participation-enabling packages (Table 10 and Table 11), on the content schema layer (Table 2), the workflow schema layer (Table 3) and the application and infrastructure layers (Table 12 and Table 4). For the analysis we choose only the last business year (namely 2014), since the participation-enabling "middle" layers developed over time as explained earlier. This vertical, multilayered view, which recognizes a coalescence of informational and executable artifacts, is also depicted in Fig. 12 (Additional file 12) for the business year 2014.

When we compare the number of contributions per modified artifact (at the top of Fig. 13 (Additional file 13)), we see that this value at the informational layer of 2.9 is higher than at the content schema and workflow schema layer. However, the values for the program artifacts are much higher. This means that the artifacts are much more frequently changed at the application and infrastructure layers. Similarly, contributors tend to modify more artifacts at the application and infrastructure layers, and the least at the content schema layer. The latter seems to come from the nature of the content schemata since these are rather small items which are relative stable. When we compare collaboration across these layers (at the bottom of Fig. 13, Additional file 13), we see that in this year the collaboration per co-artifact was actually very high, while the co-artifacts per collaborator show a similar pattern as on the contribution side. These findings demonstrate that the high number of contributions is much more due to the high number of enabled individuals rather than due to single individuals contributing very frequently.

Table 9 Collaboration at the application and infrastructure layers

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	100.0	12	59	3515	1272	9.8	10.5	585.8	0.894
2014	100.0	7	21	494	131	6.0	6.0	141.1	1.000
2013	100.0	6	15	135	82	5.0	5.0	45.0	1.000
2012	100.0	6	12	147	91	4.0	4.0	49.0	0.800
2011	85.7	6	15	268	149	5.0	5.0	89.3	1.000
2010	100.0	9	32	548	262	7.1	7.0	121.8	0.889
2009	100.0	7	14	379	300	4.0	4.0	108.3	0.667
2008	100.0	6	13	194	170	4.3	4.5	64.7	0.867

Table 10 Contributions within participation-enabling packages in the content repository

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	27315	31.0	670478	2071444	608811	24.5	12.0	75.8
2014	16421	65.5	319962	878918	302856	19.5	12.0	53.5
2013	15356	59.2	247097	629765	227836	16.1	9.0	41.0
2012	8834	31.2	47141	229579	35703	5.3	2.0	26.0
2011	3062	10.1	23805	141265	18186	7.8	1.0	46.1
2010	3465	11.7	11738	107960	8905	3.4	1.0	31.2
2009	443	1.7	5390	17154	3682	12.2	2.0	38.7
2008	311	1.2	2860	8773	2458	9.2	2.0	28.2
2007	85	0.3	644	2078	579	7.6	2.0	24.4
2006	1	0.0	1	3	1	1.0	1.0	3.0

As both the individuals and artifacts are placed in the graphs in Fig. 12 (Additional file 12), the area appears mostly grey. The contributions are mostly homogeneous, there is no overall structure visible. About a dozen power users (black craters) have contributed to a significantly higher number of artifacts. On the participation-enabling "intermediate" layers, one can see clearly that overall most contributors contribute to their "own" artifacts, while on the application and infrastructure layers, there are many artifacts which are effectively shared between the contributors.

The nodes in the social network graphs of Fig. 8 (Additional file 8) are collaborators. The information layer shows a dominant group of individuals collaborating strongly, followed by about 20 to 30 smaller teams with strong collaboration. A possible interpretation would be that the picture shows members of institutes working on shared learning resources. Here, the majority of individuals collaborate in small groups, many of them completely isolated (at the periphery). The graphics of the content schema and workflow schema layers show much stronger collaboration.

The narrow view on the collaboration of software developers on participation-enabling packages (see Table 13 and the bottom of Fig. 8, Additional file 8) still shows an aggregate collaboration graph of similarly high density (CD of 0.844), each developer on average collaborated with 7.6 other developers on these components. Over the years, this amounts to about 800 collaborations on about 200 co-artifacts in participation enabling packages at the application and infrastructure layers.

To sum up, the thorough investigation shows that individuals from the technical subsystems are active across all layers, and that business domain experts successively take over layers downward the stack: At the application and infrastructure layers, unsurprisingly, all contributions come from software developers. These software developers also contribute at the information layer, but because of the overwhelming numbers of users in the business subsystems, who predominantly contribute to this layer, this fact is practically negligible. The content schema and workflow schema layers were dominated by software developers in the early years, but this has changed in the course of time. While the software developers remain

Table 11 Collaboration on artifacts of participation-enabling packages in the content repository

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	53.0	14481	137057	257491	43271	18.9	2.0	35.6	0.001
2014	42.3	6939	15497	27073	15046	4.5	1.0	7.8	0.001
2013	46.6	7163	51759	111442	14897	14.5	1.0	31.1	0.002
2012	40.3	3563	27588	44842	7023	15.5	1.0	25.2	0.004
2011	33.7	1033	13319	24265	2646	25.8	12.0	47.0	0.025
2010	13.9	482	3271	8155	1383	13.6	7.0	33.8	0.028
2009	77.7	344	3011	6750	511	17.5	12.0	39.2	0.051
2008	55.6	173	609	880	266	7.0	4.0	10.2	0.041
2007	41.2	35	55	92	45	3.1	3.0	5.3	0.092

Table 12 Contributions to participation-enabling packages at the application and infrastructure layers

Period	Contributors	Contributor ratios	ConRels	Contributions	Artifacts	AD	MD	AWD
all	10	83.3	925	4570	463	92.5	77.0	457.0
2014	7	100.0	360	1347	193	51.4	52.0	192.4
2013	5	83.3	182	530	140	36.4	36.0	106.0
2012	6	100.0	149	534	104	24.8	25.5	89.0
2011	7	100.0	165	398	108	23.6	25.0	56.9
2010	7	77.8	196	533	114	28.0	17.0	76.1
2009	2	28.6	167	654	118	83.5	83.5	327.0
2008	3	50.0	182	221	116	60.7	68.0	73.7

active at these "intermediate" layers – in order to support the business domain experts, as the respective collaboration analysis (see Section 5.4.2) has revealed – in 2014 the majority of contributors at both participation-enabling layers are business domain experts.

6 Related work

Research efforts that aimed at understanding co-development often had a singular perspective: they considered either the technical subsystems of the business information system (e.g. analyzing source code repositories and issue trackers) or the business subsystems (e.g. investigating wiki co-authorship).

A range of research efforts puts an emphasis on the technical subsystems (which also comprises informational and social artifacts, but emphasizes the technological ones): for example, de Souza et al. [50] developed a software tool that integrates a visualization of social dependencies among developers directly into the programming environment, by analyzing dependencies among software artifacts. Sarma et al. [90] present a tool for exploring a software project through an analysis of data from its source code management system, bug tracker, and mailing list archives. Schwind et al. [91] extended a tool for network analysis of source code bases in order to measure the quality of a software developer's work. Kuk and Stevens [92] researched the impact of large corporate interests onto the democratized open source software development process. Teixeira and Lin [93] studied the collaboration on open source artifacts between large, competing enterprises. Madey et al. [94] analyzed developer collaboration networks in open source software projects. Their study represents an investigation of collective collaboration among technical stakeholders, which – solely based on joint project memberships – applies a very broad requirement for establishing collaboration relationships. Lungu [95] presents an approach to reverse engineering of ecosystems of software repositories, that touches on collaboration among developers. Hong et al. [96] investigated the evolution of large social networks of open source software developers. An example for an investigation of the evolution of a technological artifact during its co-development is provided by Pan et al. [97], who study the evolution of object-oriented software using complex network theory. Because of their much narrower focus these studies provide a relatively "deep" investigation of their respective aspects within the technical subsystems. In contrast, we aim at a more holistic view of co-development within an organization as a whole.

Similarly, there are studies which focus primarily on co-development of the business subsystems (which emphasizes informational artifacts but comprises social and technical artifacts as well). For example, Laniado et al. [98] investigated informational artifacts by studying the social and conversational structures underlying the discussions related to Wikipedia articles. An analysis of the semantic structures within Wikipedia's informational corpus, on the other hand, was conducted by Holloway et al. [99].

Laniado and Tasso [49] study collaboration patterns among co-authors within the English Wikipedia community.

Approaches for visualizing co-authorship networks include, for example, three-dimensional graph forms [100]. Kane and Alavi [51] model users and technology as a bipartite social network: they argue that while communication support systems may be visualized as edges, information management technology may be modeled as vertices.

There are studies with a more integrative perspective, too. For studying collaboration in the context of requirements engineering, Damian et al. [101] considered stakeholders from both the technical and the business subsystems and identified different types of networks (based on co-artifacts, communication, awareness, and coordinative assignment). Hence, they apply a much narrower focus on collaborative development of a specific type of informational artifacts (requirements). Frank [102] suggests a multileveled paradigm for information systems design from a meta-modeling perspective. Recently, Aram and Neumann [46] propose the vertical integration

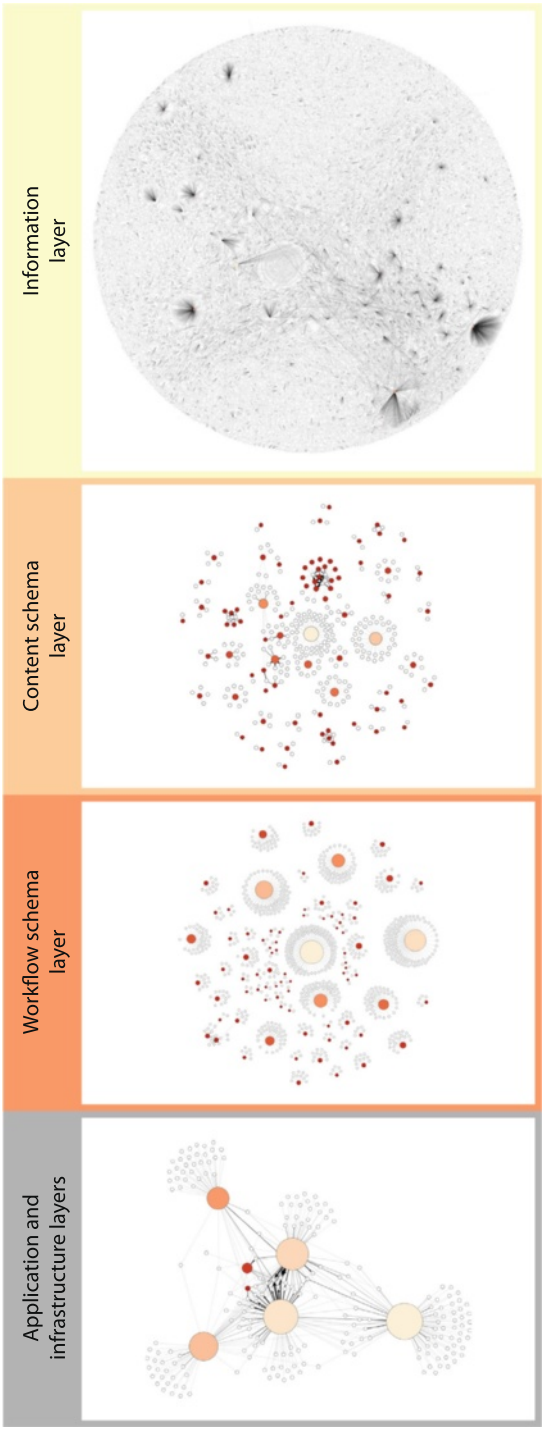

Fig. 12 A multilayered perspective of contributions to participation-enabling packages at the application and infrastructure layers, to executable specifications (content schemata and workflow schemata) at the participation-enabling layers, and to informational artifacts instantiated from artifacts at these lower layers (information layer). Vertices[1]: contributors – Vertices[2]: artifacts • Edges: contribution relationship • Rating: size of vertices[1] is related to their out-degree. Edge weights are related to contributions. • Time frame: 2014 • Layout algorithm: ForceAtlas2 [86]

of DSLs by linking multiple stakeholder perspectives via collective concept modeling.

To our best knowledge, an instrument that aims at a more detailled understanding of the participation of stakeholders in the evolutionary co-development of business information systems was missing.

7 Discussion

The main stimulus for the development of the presented approach was to develop an instrument for studying (the shift) of co-development across multiple layers of information systems. The developed approach can help researchers and practitioners in characterizing particular real-world information systems in terms of their respective co-development structures. In general, social network analysis provides an appropriate means for measuring the direct participation of business stakeholders in terms of contributions to, and co-development of, artifacts.

We have already mentioned that fostering collaboration cannot be a dogma [56], and the same is true for collective contributions. Nevertheless, the open-source and Web 2.0 movements suggest that knowledge sharing of large numbers of participants can lead to high quality information and software artifacts. Therefore, the insights gained from the study must be assessed in the light of the underlying goal of the Learn@WU system of empowering business stakeholders to participate directly in the system development.

A supremacy of the business subsystems can be expected for most (successfully adopted) business information systems. However, visualizing the dominance of business stakeholders in terms of their share of contributions to the overall system (by contrasting contribution figures in Tables 1 and 4), and their share in its collective collaboration structure (Fig. 9, Additional file 9), reinforces the arguments urged by proponents of the idea of end-user development [72]. Technological approaches such as DSLs [103], more natural environments for programming [104], or enterprise wiki systems [35] target at the coalescence of the technical and the business subsystems. Therefore, instead of treating these seemingly detached broader subsystems – which are dominated by technological artifacts and informational artifacts, respectively – as separate phenomena, we have integrated them into a common picture (see Figs. 8, Additional file 8 and 12, Additional file 12), and have taken a close look at the layers in between.

By looking at the evolution of participation (contributions and collaborations) year by year one can see how the co-development has changed over the years. According to the contribution figures (Table 1), during the last seven years the numbers of contributions, artifacts, and collaborators per year have been constantly growing. The

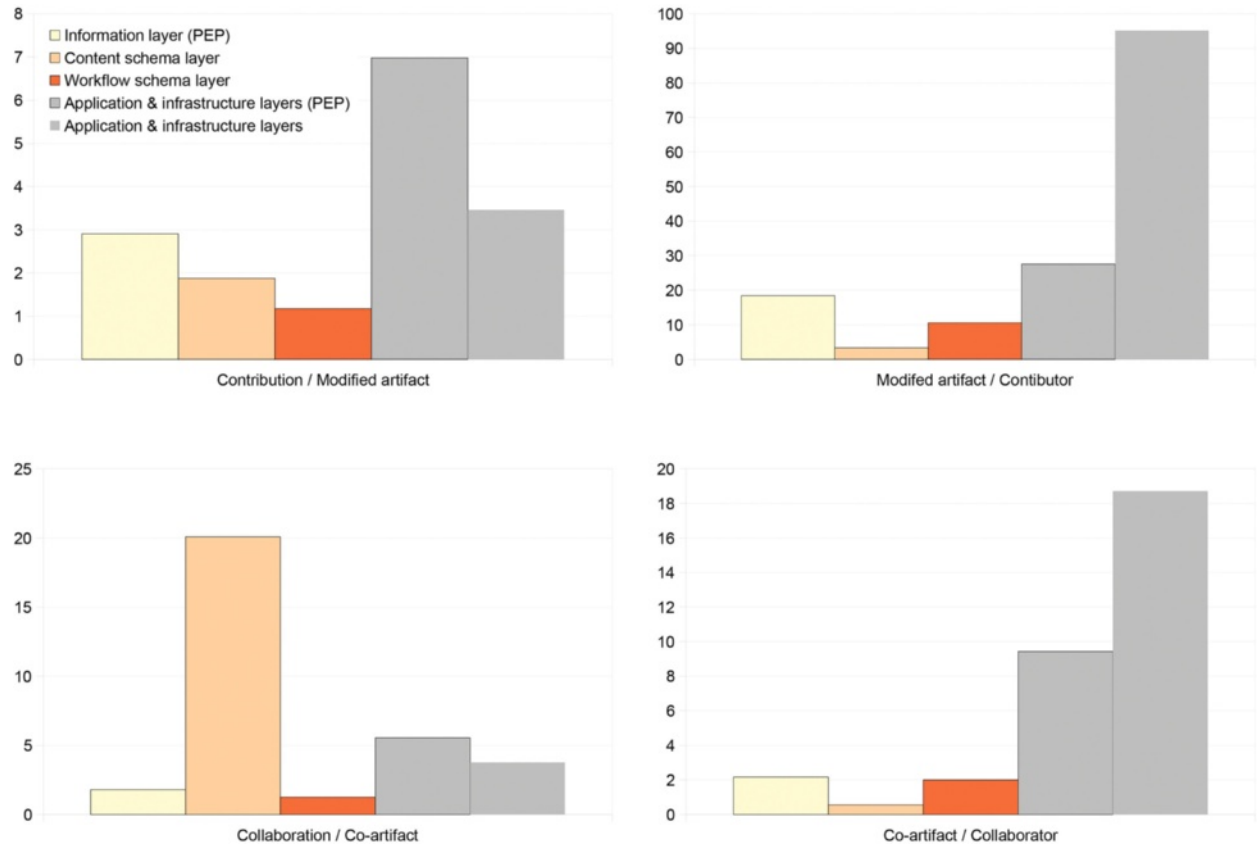

Fig. 13 Contributions and co-development of participation-enabling packages (PEP) across layers

collaboration figures (Table 6) show that the number of collaborators has also increased significantly. The collaboration ratio of over 40 % means that nearly half of the contributors have actually collaborated over the platform. Overall, the system achieves growing participation, which suggests a high perceived usefulness for the users. The rather constant number of technical contributions in the technical subsystems and the strongly increasing number of the contributor ratio and collaborator ratio by business stakeholders can be regarded as a trend. A more or less stable evolution in the technical subsystems opposes co-development patterns that constantly require refactoring in the business subsystems.

We see growing numbers of contributors across these layers (illustrated in Fig. 12, Additional file 12): compared to the application layer, there were about 10 times as many contributors active at the participation-enabling layers, and about 2,000 times as many at the information layer. At the workflow schema layer, we see growing numbers of contributors, artifacts and contributions over the years

Table 13 Collaboration on participation-enabling packages at the application and infrastructure layers

Period	Collaborator ratio	Collaborators	ColRels	Collaborations	Co-artifacts	CAD	CMD	CAWD	CD
all	100.0	10	38	879	236	7.6	8.0	175.8	0.844
2014	100.0	7	21	366	66	6.0	6.0	104.6	1.000
2013	100.0	5	9	51	35	3.6	4.0	20.4	0.900
2012	100.0	6	8	55	35	2.7	3.0	18.3	0.533
2011	85.7	6	12	81	39	4.0	4.0	27.0	0.800
2010	100.0	7	15	111	56	4.3	5.0	31.7	0.714
2009	100.0	2	1	49	49	1.0	1.0	49.0	1.000
2008	66.7	2	1	66	66	1.0	1.0	66.0	1.000

(Fig. 5 (Additional file 5), Table 3). One can see that part of the system development is already shifting from the software developers up to business domain experts. The analysis of the contributions and collaborations shows that the system succeeded to enable business experts to effectively contribute to the system. The degree of co-development in the overall system increased significantly over the years.

The social network analysis shows, that in contrast to the contribution figures, the collaboration in terms of group size (degree metrics) tends to decrease bottom up across layers; the collaboration at the participation-enabling layers shows a tendency towards small groups of collaborators (Tables 7 and 8), but the median degree of collaborations still tends to be larger than in the overall business subsystems. However, overall the work on executable specifications, and in particular on workflow schemata, are currently more the product of individual work than that of team efforts. This should not be misconstrued as an undesirable evolution, as, for example, Bird et al. [52] showed that lower levels of artifact ownership (more contributors) can correlate with diminished quality.

Limitations and research directions. There are several more or less obvious limitations of our study, which point directions for future research.

In our application study, we employ a technological approach, where we only captured direct contributions to digital artifacts. Any information flow within the system that is not (yet) mediated by software technology remains concealed. This is not a methodological limitation, since a more complete investigation of the system based on qualitative studies about the more strategic organizational layers could be conducted. Furthermore, measuring direct contributions can bias authorship, e.g. when somebody commits a patch by someone else. However, since we are not interested in measuring individual contributions but contributions by groups of similar stakeholders, this kind of limitation is of less importance.

Secondly, our main contribution is an approach to systematic investigation of the co-development of information systems. Further applications of our approach may, of course, delve deeper in many aspects. One could consider to differentiate the stakeholders in the business subsystems according to their respective roles (students/teachers/staff). Similarly, a comparison of the organizational chart (the "planned structures") with the actual collaboration structures [105] would provide deeper insights with respect to the observed phenomena. Taxonomies could be applied to the corpus of artifacts, in order to identify individual, semantically linked instances across layers (e.g. for connecting a natural language learning instruction (information layer), to its template (content schema layer), its learning script [31] (workflow schema layer), and its application layer component). Similarly, we have not assessed the semantics and quality attributes of contributions. We have not considered properties such as the correctness, size, longevity, or impact of contributions. Burnett [106] reflects on software quality issues in the context of end-user software engineering. The concepts behind change bursts [107], which allow to predict defects based on sequences of contributions, might be applicable to artifacts across layers. Concepts such as code ownership [52], and intellectual authorship in general could be incorporated. Also, a more fine-grained investigation with respect to the collaboration structures would allow to differentiate e.g. more intense forms of co-development, such as coordinated collaboration.

In larger organizations with a detailed separation of labor reflected in the organization structure, the metaphor of social networks appears well-suited for analysis. Similarly, it is well-suited for studying the contributions and collaboration structures of such organizations. Instruments for conducting analyses of such networks provide indispensable means for both research and practice. The field of software engineering can benefit from these influences particularly in the areas of collaborative large-scale software development, and end-user development [108]. In the context of studies such as the one presented in this paper, social network analysis helps to characterize the participation structures at the different layers and to contrast these with the other layers. For example, the different value ranges of the layers with respect to the degree-based metrics distinctly characterize the social networks at these layers (e.g., compare Tables 1, 2 and 4). Particularly when supported by visualizations, these analysis techniques also help in identifying communities with strong actual co-development behavior (see e.g. the easily distinguishable collaboration clusters within the business subsystems in Fig. 9 (Additional file 9)). However, the interpretation of these graphs (how certain clusters relate to projects or groups) requires domain and organizational knowledge. In this sense the resulting graphs cannot provide a full picture of the contribution and collaboration structures, but provide rather a means to detect structures in highly complex graphs that possibly require deeper investigations of the observed phenomena. Further investigations could focus on detecting [109] and qualitatively analyzing [110] these communities in the network. Such insights can be used to trigger further participation-enabling measures, e.g. the development of task-specific languages for these groups.

While we could show that we could apply social network analysis to study collaboration structures in fairly large information systems (analyzing millions of artifacts and contributions by ten-thousands of contributors), it became evident that a single-layer analysis hides

a lot of interesting details. The vast amount of contributions at the informational layer dominates all kinds of visualizations. The application of the multi-layered approach helped us to understand how the contribution and collaboration structures developed, thus providing insights on a more general level.

A successful application of the approach, i.e. to conduct a study such as the one presented in this paper, depends on certain circumstances and properties of the system. In our case we could exploit a code and content repository for mining contributions to up to 10 years. For the analysis of other business information systems, obtaining contribution data might be prohibitive expensive. Hence, for successfully investigating the continuous co-development structures within a business information system, those responsible should strive for the following idealistic situation: actual and potential contributors should be uniquely identifiable across subsystems and layers, e.g. via an organization-wide centralized identity management and authentication. Artifacts across layers should be managed technically as coherently as possible, ideally within an overarching system-wide object system. All contributions to artifacts should be manifested and traceable, e.g. in the form of object revisions. Ideally, artifacts and contributions would be enriched with semantic metadata. A system-wide log of deliberate co-development-enabling measures should be maintained.

8 Conclusion

In this paper, we have presented a novel approach that facilitates the attainment of a comprehensive overview of co-development patterns within existing, situated information systems. The approach incorporates a multi-layered perspective that explicitly recognizes the co-development of the system by business domain experts. We have demonstrated the utility of our approach in the context of a real-world educational business information system. The case study revealed a strong and growing dominance of business domain experts and end users in terms of their share in the co-development of the overall system over the years. This increasing direct participation suggests both a high perceived usefulness of the system and a successful step-wise provisioning of participation-enabling end-user applications. The application of the multi-layered approach and the identification and analysis of the participation-enabling layers facilitated the understanding of the co-development structures within the system. We believe that the presented approach can help to support both researchers and practitioners in revealing existing structures of co-development within an information system and in evaluating the impact of measures taken to foster co-development.

Additional files

Additional file 1: Business information system. This contrived visualization of a business information system as a complex compound artifact illustrates its interconnected inner subsystems interwoven with its outer environments in the form of a bipartite network comprising artifacts and individuals.

Additional file 2: Different co-development structures in business information systems: business domain experts can contribute to executable artifacts either only indirectly (case a) or also directly (case b).

Additional file 3: Layering.

Additional file 4: A layered perspective of the Learn@WU technology stack.

Additional file 5: Evolution of contributions at participation-enabling layers. x-axes: business years • y-axes: logarithmic scale of contributors/contributions.

Additional file 6: Contributions in the technical subsystems at the application and infrastructure layers. Vertices [1]: 12 software developers – Vertices [2]: 9,062 artifacts (mostly source code) • Edges: contribution relationship • Rating: size of vertices [1] is related to their out-degree. Edge weights are related to contributions. • Time frame: 2008–2014 • Layout algorithm: ForceAtlas2 [86].

Additional file 7: Evolution of contributions in the technical subsystems at the application and infrastructure layers. Vertices [1]: software developers – Vertices [2]: artifacts (mostly source code) • Edges: contribution relationship • Rating: size of vertices [1] is related to their out-degree. Edge weights are related to contributions. • Time frame: 2008–2014 • Layout algorithm: ForceAtlas2 [86].

Additional file 8: A multilayered perspective of collaboration on and within participation-enabling packages in 2014. Vertices: collaborators • Edges: collaboration relationships • Rating: size and color saturation of vertices are related to weighted degree. Edge weights are related to collaboration intensity. • Time frame: 2014 • Layout algorithm: ForceAtlas2 [86].

Additional file 9: Collective collaboration within Learn@WU, among business and technical stakeholders, respectively. Vertices: 15,180 collaborators in the business subsystems, 12 collaborators in the technical subsystems • Edges: collaboration relationships • Rating: size and color saturation of vertices are related to weighted degree. Edge weights are related to collaboration intensity. • Time frame: 2004–2014 (business subsystems), 2008–2014 (technical subsystems) • Layout algorithm: ForceAtlas2 [86].

Additional file 10: Evolution of collaboration in Learn@WU, among business and technical stakeholders, respectively. The figure aggregates unipartite graphs that show the collaboration relations (edges) between collaborating stakeholders (vertices) in the Learn@WU system. Size and color of both vertices and edges relate to weighted degree/weight within each graph. Layout algorithms: ForceAtlas2 [86] (business subsystems) and Fruchterman-Reingold [87] (technical subsystems).

Additional file 11: Evolution of collaborations at participation-enabling layers. x-axes: business years • y-axes: logarithmic scale of collaborators/collaborations.

Additional file 12: A multilayered perspective of contributions to participation-enabling packages at the application and infrastructure layers, to executable specifications (content schemata and workflow schemata) at the participation-enabling layers, and to informational artifacts instantiated from artifacts at these lower layers (information layer). Vertices[1]: contributors – Vertices[2]: artifacts • Edges: contribution relationship • Rating: size of vertices[1] is related to their out-degree. Edge weights are related to contributions. • Time frame: 2014 • Layout algorithm: ForceAtlas2 [86].

Additional file 13: Contributions and co-development of participation-enabling packages (PEP) across layers.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
Both authors have equally contributed to the literature review, the theoretical and conceptual contribution, the research design, the analysis, and the text. Both authors have read and approved the final manuscript.

Authors' information
GN is the chair of the Institute for Information Systems and New Media of the Vienna University of Economics and Business. MA is doctoral candidate of information systems at this institute.

Acknowledgements
The authors thank the editor and the referees for their helpful and constructive comments throughout the whole review process. This profound feedback has lead to a significantly improved presentation of our work.

Received: 15 October 2014 Accepted: 8 June 2015
Published online: 01 July 2015

References

1. Lewin AY, Long CP, Carroll TN (1999) The coevolution of new organizational forms. Organ Sci 10(5):535–550
2. Bjerknes G, Bratteteig T, Espeseth T (1991) Evolution of finished computer systems. Scand J Inform Syst 3:25–45
3. Hippel Ev (2005) Democratizing innovation: The evolving phenomenon of user innovation. J für Betriebswirtschaft 55(1):63–78. doi:10.1007/s11301-004-0002-8
4. Neumann G, Sobernig S, Aram M (2014) Evolutionary business information systems: Perspectives and challenges of an emerging class of information systems. Bus Inform Syst Eng 6(1):33–38. doi:10.1007/s12599-013-0305-1
5. Strogatz SH (2001) Exploring complex networks. Nature 410(6825):268–276. doi:10.1038/35065725
6. Wasserman S, Faust K (1994) Social Network Analysis: Methods and Applications. 1st edn.. Cambridge University Press, Cambridge, New York
7. Newman MEJ (2001) The structure of scientific collaboration networks. Proc Nat Acad Sci 98(2):404–409. doi:10.1073/pnas.021544898
8. Moreno JL (1934) Who Shall Survive: A New Approach to the Problem of Human Interrelations. Nervous and Mental Disease Monograph Series, Vol. 58. Nervous and Mental Disease Publishing Co, Washington, DC, USA
9. Tichy NM, Tushman ML, Fombrun C (1979) Social network analysis for organizations. Acad Manag Rev 4(4):507–519. doi:10.2307/257851
10. Lim SL, Quercia D, Finkelstein A (2010) StakeNet: Using social networks to analyse the stakeholders of large-scale software projects. In: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1. ACM, New York, NY, USA. pp 295–304. doi:10.1145/1806799.1806844
11. Freeman LC (1978) Centrality in social networks - conceptual clarification. Soc Netw 1(3):215–239. doi:10.1016/0378-8733(78)90021-7
12. Scott J (2012) Social Network Analysis. 3rd edn.. Sage Publications Ltd, Los Angeles
13. Sommerville I (2010) Software Engineering. 9th revised edition edn.. Addison-Wesley Longman, Amsterdam
14. Bourque P, Fairley RED (eds) (2014) SWEBOK V3.0 - Guide to the Software Engineering Body of Knowledge. IEEE Computer Society, Piscataway, NJ
15. Bryant BR, Gray J, Mernik M (2010) Domain-specific software engineering. In: Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research. ACM, New York, NY, USA. pp 65–68. doi:10.1145/1882362.1882376
16. Mernik M, Heering J, Sloane AM (2005) When and how to develop domain-specific languages. ACM Comput Surv 37(4):316–344. doi:10.1145/1118890.1118892
17. Ko AJ, Myers B, Rosson MB, Rothermel G, Shaw M, Wiedenbeck S, Abraham R, Beckwith L, Blackwell A, Burnett M, Erwig M, Scaffidi C, Lawrance J, Lieberman H (2011) The state of the art in end-user software engineering. ACM Comput Surv 43(3):1–44. doi:10.1145/1922649.1922658
18. Hansen HR, Neumann G (2009) Wirtschaftsinformatik 1. 10edn.. UTB, Stuttgart
19. Orlikowski WJ, Iacono CS (2001) Research commentary: Desperately seeking the IT in IT research—a call to theorizing the IT artifact. Inform Syst Res 12(2):121–134. doi:10.1287/isre.12.2.121.9700
20. Conen W, Neumann G (1998) A perspective on technology-assisted collaboration. In: Conen W, Neumann G (eds). Coordination Technology for Collaborative Applications. Lecture Notes in Computer Science. Springer. pp 1–7
21. Hevner AR, March ST, Park J, Ram S (2004) Design science in information systems research. MIS Quarterly 28(1):75–105
22. Lee AS (2010) Retrospect and prospect: information systems research in the last and next 25 years. J Inform Technol 25(4):336–348. doi:10.1057/jit.2010.24
23. Lee A, Thomas M, Baskerville R (2013) Going back to basics in design: From the IT artifact to the IS artifact. In: Proceedings of the 19th Americas Conf. on Inf. Systems, Chicago, Illinois
24. Simon HA (1996) The Sciences of the Artificial. 3rd edn.. MIT Press, Cambridge, MA, USA
25. Popper K (1980) Three worlds. The Tanner Lecture on Human Values, Vol. 1. Cambridge University Press, Cambridge, UK. http://tannerlectures.utah.edu/lecture-library.php
26. Vahidov R (2006) Design researcher's IS artifact: a representational framework. In: Proceedings of the 1st International Conference on Design Science Research in Information Systems and Technology, Claremont, USA, Claremont, CA
27. McKinney EHJ, Yoos CJI (2010) Information about information: A taxonomy of views. MIS Quarterly 34(2):329–344
28. Germonprez M, Hovorka D, Gal U (2011) Secondary design: A case of behavioral design science research. J Assoc Inform Syst 12(10):662–683
29. March ST, Smith GF (1995) Design and natural science research on information technology. Decis Support Syst 15(4):251–266. doi:10.1016/0167-9236(94)00041-2
30. Spinellis D (2001) Notable design patterns for domain-specific languages. J Syst Softw 56(1):91–99. doi:10.1016/S0164-1212(00)00089-3
31. Weinberger A, Ertl B, Fischer F, Mandl H (2005) Epistemic and social scripts in computer-supported collaborative learning. Instr Sci 33(1):1–30. doi:10.1007/s11251-004-2322-4. Accessed 2014-06-15
32. Strembeck M, Zdun U (2009) An approach for the systematic development of domain-specific languages. Softw Prac Exp 39(15):1253–1292. doi:10.1002/spe.936
33. Fowler M (2010) Domain-Specific Languages. 1st edn.. Addison-Wesley Professional, Westford, Massachusetts
34. Kelleher C, Pausch R (2005) Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. ACM Comput Surv 37(2):83–137. doi:10.1145/1089733.1089734. Accessed 2015-01-15
35. Stocker A, Tochtermann K (2011) Enterprise wikis – types of use, benefits and obstacles: A multiple-case study. In: Fred A, Dietz JLG, Liu K, Filipe J (eds). Knowledge Discovery, Knowlege Engineering and Knowledge Management. Communications in Computer and Information Science. Springer, Berlin Heidelberg. pp 297–309
36. Pahlke I, Beck R, Wolf M (2010) Enterprise mashup systems as platform for situational applications. Bus Inform Syst Eng 2(5):305–315. doi:10.1007/s12599-010-0121-9
37. Germonprez M, Hovorka D, Collopy F (2007) A theory of tailorable technology design. J Assoc Inform Syst 8(6):351–367
38. Object Management Group (2014) Essence – kernel and language for software engineering methods. beta 2. Technical report, Object Management Group, Inc. (OMG), Needham, MA, USA. http://www.omg.org/spec/Essence/1.0/
39. Dingsøyr T, Dybå T, Moe NB (eds) (2010) Agile Software Development: Current Research and Future Directions. Springer, Berlin
40. Muller MJ, Kuhn S (1993) Participatory design. Commun ACM 36(4):24–28
41. Barki H, Hartwick J (1994) Measuring user participation, user involvement, and user attitude. MIS Quarterly 18(1):59. doi:10.2307/249610
42. Lim SL, Bentley PJ (2011) Evolving relationships between social networks and stakeholder involvement in software projects. In: Proceedings of the

13th Annual Conference on Genetic and Evolutionary Computation. ACM, New York, NY, USA. pp 1899–1906. doi:10.1145/2001576.2001831
43. Nagappan N, Murphy B, Basili V (2008) The influence of organizational structure on software quality: An empirical case study. In: Proceedings of the 30th International Conference on Software Engineering. ACM, New York, NY, USA. pp 521–530. doi:10.1145/1368088.1368160
44. Bird C, Nagappan N, Devanbu P, Gall H, Murphy B (2009) Does distributed development affect software quality?: an empirical case study of windows vista. Commun ACM 52(8):85–93. Accessed 2015-01-15
45. West J, O'Mahony S (2005) Contrasting community building in sponsored and community founded open source projects. IEEE, Washington, DC, USA. pp 196–205. doi:10.1109/HICSS.2005.166
46. Aram M, Neumann G (2014) Exploring collective DSL integration in a large situated IS: Towards comprehensive language integration in information systems. In: Proceedings of the 2014 European Conference on Software Architecture Workshops. ACM, New York, NY, USA. pp 20–1205. doi:10.1145/2642803.2642823
47. Aram M (2013) Towards a framework for analytics-driven domain-specific mashup environments. In: Proceedings of the Doctoral Consortium at the European Conference on Technology Enhanced Learning 2013, Paphos, Cyprus. pp 1–6
48. Shneiderman B (1981) Direct manipulation: A step beyond programming languages. In: ACM SIGSOC Bulletin. ACM, New York, NY, USA Vol. 13. p 143
49. Laniado D, Tasso R (2011) Co-authorship 2.0: Patterns of collaboration in wikipedia. In: Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia. ACM, New York, NY, USA. pp 201–210. doi:10.1145/1995966.1995994
50. de Souza CR, Quirk S, Trainer E, Redmiles DF (2007) Supporting collaborative software development through the visualization of socio-technical dependencies. In: Proceedings of the 2007 International ACM Conference on Supporting Group Work. GROUP '07. ACM, New York, NY, USA. pp 147–156. doi:10.1145/1316624.1316646. http://doi.acm.org/10.1145/1316624.1316646 Accessed 2014-12-05
51. Kane GC, Alavi M (2008) Casting the net: A multimodal network perspective on user-system interactions. Inform Syst Res 19(3):253–272. doi:10.1287/isre.1070.0158
52. Bird C, Nagappan N, Murphy B, Gall H, Devanbu P (2011) Don't touch my code!: examining the effects of ownership on software quality. In: Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering. ACM. pp 4–14. http://dl.acm.org/citation.cfm?id=2025119 Accessed 2014-12-06
53. Briggs RO, Kolfschoten G, Vreede GqJd, Albrecht C, Dean DR, Lukosch S (2009) A seven-layer model of collaboration: Separation of concerns for designers of collaboration systems. In: Proceedings of the International Conference on Information Systems. ICIS, Phoenix, Arizona, USA. http://aisel.aisnet.org/icis2009/26/
54. Nunamaker JF, Briggs RO, Vreede GJ (2001) From information technology to value creation technology. In: Dickson GW, DeSanctis G (eds). Information Technology and the Future Enterprise. Prentice Hall, New-York
55. de Vreede G, Briggs RO (2005) Collaboration engineering: Designing repeatable processes for high-value collaborative tasks. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005. HICSS '05. pp 17–17. doi:10.1109/HICSS.2005.144
56. Hansen MT (2009) When internal collaboration is bad for your company. Harv Bus Rev 87(4):82–88. Accessed 2014-12-05
57. Cross RL, Parker A (2004) Hidden Power of Social Networks: Understanding How Work Really Gets Done in Organizations. Auflage: New. edn. Harvard Business Review Press, Boston, Mass
58. Sowe S, Stamelos I, Angelis L (2006) Identifying knowledge brokers that yield software engineering knowledge in OSS projects. Inform Softw Technol 48(11):1025–1033. doi:10.1016/j.infsof.2005.12.019. Accessed 2014-09-11
59. Conaldi G, Lomi A (2013) The dual network structure of organizational problem solving: A case study on open source software development. Soc Netw 35(2):237–250. doi:10.1016/j.socnet.2012.12.003. Accessed 2015-01-09
60. Andriotis P, Tzermias Z, Mparmpaki A, Ioannidis S, Oikonomou G (2013) Multilevel visualization using enhanced social network analysis with smartphone data. Int J Digit Crime For 5(4):34–54. doi:10.4018/ijdcf.2013100103. Accessed 2014-09-07
61. de Souza C, Froehlich J, Dourish P (2005) Seeking the source: Software source code as a social and technical artifact. In: Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work. GROUP '05. ACM, New York, NY, USA. pp 197–206. doi:10.1145/1099203.1099239. http://doi.acm.org/10.1145/1099203.1099239 Accessed 2015-01-09
62. Van Der Aalst WMP, Reijers HA, Song M (2005) Discovering social networks from event logs. Comput Supported Coop Work 14(6):549–593. doi:10.1007/s10606-005-9005-9. Accessed 2015-01-09
63. Alberer G, Alberer P, Enzi T, Ernst G, Mayrhofer K, Neumann G, Rieder R, Simon B (2003) The learn@WU learning environment. In: Wirtschaftsinformatik 2003/Band I. Physica-Verlag, Heidelberg. pp 593–612. doi:10.1007/978-3-642-57444-3_31
64. Vienna University of Economics and Business. http://www.wu.ac.at Accessed 2014-09-28
65. Frank U Towards a pluralistic conception of research methods in information systems research. ICB Research Reports 7, University Duisburg-Essen, Institute for Computer Science and Business Information Systems (ICB), Duisburg-Essen
66. Österle H, Becker J, Frank U, Hess T, Karagiannis D, Krcmar H, Loos P, Mertens P, Oberweis A, Sinz EJ (2010) Memorandum on design-oriented information systems research. Eur J Inform Syst 20(1):7–10. doi:10.1057/ejis.2010.55
67. Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2007) A design science research methodology for information systems research. J Manag Inform Syst 24(3):45–77. doi:10.2753/MIS0742-1222240302
68. Benbasat I, Goldstein DK, Mead M (1987) The case research strategy in studies of information systems. MIS Quarterly 11(3):369–386. doi:10.2307/248684
69. Lee AS, Baskerville RL (2003) Generalizing generalizability in information systems research. Inform Syst Res 14(3):221–243. doi:10.1287/isre.14.3.221.16560
70. Baskerville R, Pries-Heje J, Venable J (2009) Soft design science methodology. In: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology. DESRIST '09. ACM, New York, NY, USA. pp 9–1911. doi:10.1145/1555619.1555631. http://doi.acm.org/10.1145/1555619.1555631
71. OpenACS. http://openacs.org Accessed 2014-10-01
72. Sutcliffe A, Mehandjiev N (2004) End-user development. Commun ACM 47(9):31–32. doi:10.1145/1015864.1015883
73. MacLean A, Carter K, Lövstrand L, Moran T (1990) User-tailorable systems: pressing the issues with buttons. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, New York, NY, USA. pp 175–182. doi:10.1145/97243.97271
74. Wu F, Weld DS (2008) Automatically refining the wikipedia infobox ontology. In: Proceedings of the 17th International Conference on World Wide Web. ACM, New York, NY, USA. pp 635–644. doi:10.1145/1367497.1367583
75. LRN - Learn, Research, Network. http://dotlrn.org/
76. Blesius CR, Moreno-Ger P, Neumann G, Raffenne E, Boticario JG, Kloos CD (2007) LRN: E-learning inside and outside the classroom. In: Fernández-Manjón B, Sánchez-Pérez JM, Gómez-Pulido JA, Vega-Rodríguez MA, Bravo-Rodríguez J (eds). Computers and Education. Springer, Netherlands. pp 13–25. http://goo.gl/VuPcpa
77. Neumann G, Erol S (2009) From a social wiki to a social workflow system. In: Ardagna D, Mecella M, Yang J (eds). Business Process Management Workshops. Lecture Notes in Business Information Processing. Springer, Berlin Heidelberg. pp 698–708
78. PostgreSQL. http://www.postgresql.org Accessed 2014-10-02
79. NaviServer. http://naviserver.sourceforge.net Accessed 2014-09-26
80. Hernández R, Grumet A (2005) OpenACS: robust web development framework. In: Proceedings of the Tcl/Tk 2005 Conference, Portland, Oregon, Portland, Oregon
81. Demetriou N, Koch S, Neumann G (2006) The development of the OpenACS community. In: Lytra M, Naeve A (eds). Open Source for Knowledge and Learning Management: Strategies Beyond Tools. Idea Group Publishing, Hershey, PA

82. WU Annual Reports. http://goo.gl/ItV5iC Accessed 2014-09-29
83. Tyler J, Cheikes B, Farance F, Tonkel J (2003) 1484.1 - IEEE standard for learning technology - learning technology systems architecture (LTSA). Technical report, IEEE Computer Society, New York
84. Torvalds L, Hamano J (2010) Git: Fast version control system. http://git-scm.com
85. Gephi. http://www.gephi.org Accessed 2014-09-25
86. Jacomy M, Venturini T, Heymann S, Bastian M (2014) ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PLOS ONE 9(6):1–12. doi:10.1371/journal.pone.0098679
87. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. Softw Prac Exp 21(11):1129–1164. doi:10.1002/spe.4380211102
88. Cypher A, Dontcheva M, Lau T, Nichols J (2010) No Code Required: Giving Users Tools to Transform the Web. Morgan Kaufmann, Burlington, USA
89. Hossain L, Wu A, Chung KKS (2006) Actor centrality correlates to project based coordination. In: Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work. CSCW '06. ACM, Banff, Alberta, Canada. pp 363–372. doi:10.1145/1180875.1180930
90. Sarma A, Maccherone L, Wagstrom P, Herbsleb J (2009) Tesseract: Interactive visual exploration of socio-technical relationships in software development. In: IEEE 31st International Conference on Software Engineering, 2009. ICSE 2009. pp 23–33. doi:10.1109/ICSE.2009.5070505
91. Schwind M, Schenk A, Schneider M (2010) A tool for the analysis of social networks in collaborative software development. In: Proceedings of the 2010 43rd Hawaii International Conference on System Sciences. HICSS '10. IEEE Computer Society, Washington, DC, USA. pp 1–10. doi:10.1109/HICSS.2010.40. http://dx.doi.org/10.1109/HICSS.2010.40 Accessed 2014-12-05
92. Kuk G, Stevens G (2010) Corporatizing open source software innovation in the plone community. In: Proceedings of the Sixteenth Americas Conference on Information Systems, Lima, Peru
93. Teixeira J, Lin T (2014) Collaboration in the open-source arena: the webkit case. In: Proceedings of the 52nd ACM Conference on Computers and People Research. ACM, New York, NY, USA. pp 121–129. doi:10.1145/2599990.2600009
94. Madey G, Freeh V, Tynan R (2002) The open source software development phenomenon: An analysis based on social network theory. Association for Information Systems, Dallas, TX
95. Lungu MF Reverse engineering software ecosystems. PhD thesis, University of Lugano
96. Hong Q, Kim S, Cheung SC, Bird C (2011) Understanding a developer social network and its evolution. In: Proceedings of the 27th IEEE International Conference on Software Maintenance (ICSM). IEEE, Williamsburg, Virginia, USA. pp 323–332. doi:10.1109/ICSM.2011.6080799
97. Pan W, Li B, Ma Y, Liu J (2011) Multi-granularity evolution analysis of software using complex network theory. J Syst Sci Complexity 24(6):1068–1082. doi:10.1007/s11424-011-0319-z
98. Laniado D, Tasso R, Volkovich Y, Kaltenbrunner A (2011) When the wikipedians talk: Network and tree structure of wikipedia discussion pages. In: Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, Barcelona, Spain. pp 177–184
99. Holloway T, Bozicevic M, Börner K (2007) Analyzing and visualizing the semantic coverage of wikipedia and its authors: Research articles. Complexity 12(3):30–40. doi:10.1002/cplx.v12:3
100. Biuk-Aghai RP (2006) Visualizing co-authorship networks in online wikipedia. In: Proceedings of the International Symposium on Communications and Information Technologies. IEEE, Bangkok. pp 737–742. doi:10.1109/ISCIT.2006.339838
101. Damian D, Kwan I, Marczak S (2010) Requirements-driven collaboration: Leveraging the invisible relationships between requirements and people. In: Mistrík I, Grundy J, Hoek A, Whitehead J (eds). Collaborative Software Engineering. Springer. pp 57–76
102. Frank PDU (2014) Multilevel modeling. Business & Information Systems Engineering 6(6):319–337. doi:10.1007/s12599-014-0350-4. Accessed 2014-11-27
103. van Deursen A, Klint P, Visser J (2000) Domain-specific languages: An annotated bibliography. ACMSIGPLAN Notices 35(6):26–36. doi:10.1145/352029.352035
104. Myers BA, Pane JF, Ko A (2004) Natural programming languages and environments. Commun ACM 47(9):47–52. doi:10.1145/1015864.1015888
105. Kadushin C (2012) Understanding Social Networks: Theories, Concepts, and Findings. Oxford University Press, New York
106. Burnett M (2009) What is end-user software engineering and why does it matter? In: End-User Development. Springer. pp 15–28
107. Nagappan N, Zeller A, Zimmermann T, Herzig K, Murphy B (2010) Change bursts as defect predictors. In: 2010 IEEE 21st International Symposium on Software Reliability Engineering (ISSRE). pp 309–318. doi:10.1109/ISSRE.2010.25
108. (2013) End-User Development. Lecture Notes in Computer Science(Dittrich Y, Burnett M, Mørch A, Redmiles D, Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, eds.), Vol. 7897. Springer, Berlin, Heidelberg. http://link.springer.com/10.1007/978-3-642-38706-7 Accessed 2015-01-10
109. Ye Q, Wu B, Wang B (2013) Detecting communities in massive networks efficiently with flexible resolution. In: Özyer T, Rokne J, Wagner G, Reuser AHP (eds). The Influence of Technology on Social Network Analysis and Mining. Lecture Notes in Social Networks. Springer, Vienna. pp 373–392
110. Moser C, Groenewegen P, Huysman M (2013) Extending social network analysis with discourse analysis: Combining relational with interpretive data. In: Özyer T, Rokne J, Wagner G, Reuser AHP (eds). The Influence of Technology on Social Network Analysis and Mining. Lecture Notes in Social Networks. Springer, Vienna. pp 547–561