Major Variants of the SIS Architecture Pattern for Collective Intelligence Systems

ANGELIKA MUSIL, JUERGEN MUSIL and STEFAN BIFFL,
Institute of Software Technology and Interactive Systems, TU Wien, Austria

Collective Intelligence Systems (CIS), such as social networking services, wikis, and media sharing platforms, access and harness the collective knowledge of connected people by providing a web-based environment to share, distribute, and retrieve topic-specific information in an efficient way. In order to design well-tailored CIS, software architects need a complete understanding about (1) architectural principles that all kinds of CIS have in common, and (2) system variants in the field. Thus to provide consolidated systematic knowledge of architectural commonalities and variations in the CIS domain, we present in this work five major pattern variants of CIS. We investigated a number of CIS in the field with focus on a detailed survey of existing variants among key architecture-significant principles based on previously identified basic concepts, principles, and characteristics of software architectures that all CIS have in common. The variants are identified along two dimensions with respect to the relationship between the key elements of artifacts and actor records across and within two layers of the system.

Categories and Subject Descriptors: D.2.11 [**Software Engineering**]: Software Architectures—*Patterns* and *Domain-specific architectures*; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human information processing*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Current awareness systems* and *Information networks*

Additional Key Words and Phrases: Collective intelligence, software architecture pattern, stigmergic information system, variability

ACM Reference Format:

Angelika Musil, Juergen Musil, and Stefan Biffl. 2016. Major Variants of the SIS Architecture Pattern for Collective Intelligence Systems. EuroPLoP '16: Proceedings of the 21st European Conference on Pattern Languages of Programs (July 2016), 11 pages.
DOI: http://dx.doi.org/10.1145/3011784.3011818

1. INTRODUCTION

Collective Intelligence Systems (CIS) like social networking services, social media sharing platforms and wikis have substantially influenced how people share and access information and knowledge. Since the emergence of CIS in the late 90's a variety of CIS have been invented, so that these systems range from team size (e.g., issue trackers) up to society-scale (e.g., YouTube). Independently of its scale, a CIS is a type of *socio-technical system*, where a web-based environment [Gruber 2008] enables a user community to share, distribute and retrieve topic-specific information in a more efficient way [Omicini and Contucci 2013; Miorandi et al. 2014]. This environment acts as a mediator [Weyns et al. 2007] of interactions among the user community and thereby provides effective, bottom-up collabora-

This work was supported by the Christian Doppler Forschungsgesellschaft, the Federal Ministry of Science, Research and Economy, the Austrian National Foundation for Research, Technology and Development, and TU Wien research funds.
Author's address: A. Musil, J. Musil, S. Biffl, Institute of Software Technology and Interactive Systems, TU Wien, Favoritenstrasse 9/188, 1040 Vienna, Austria; email: {angelika, jmusil}@computer.org, stefan.biffl@tuwien.ac.at

EuroPLoP '16, July 06 - 10, 2016, Kaufbeuren, Germany

ACM 978-1-4503-4074-8/16/07...$15.00
DOI: http://dx.doi.org/10.1145/3011784.3011818

tion and communication capabilities that rely on user-generated content, facilitating the aggregation and dissemination of knowledge in a coordinated way [Musil et al. 2015c]. Concurrently to CIS, additional systems and approaches have been established which also rely on crowds and user communities. These are crowdsourcing-like approaches [Simperl 2015] like actual crowdsourcing [Howe 2006], human computation [Ahn 2005], citizen-science in particular and social computing [Parameswaran and Whinston 2007] and platform-mediated work in general.

Although CIS have seen wide adoption through society, the software architectural knowledge about the CIS domain, that goes beyond implementation, engineering and technical architecture, remains still a topic of research. A particular direction here is to have a common understanding about the basic commonalities and variants in the domain of CIS. Therefore to design well-tailored systems, software architects need a complete understanding about (1) architectural principles that all kinds of CIS have in common, and (2) system variants in the field. Without this knowledge it is difficult for software architecture researchers and practitioners to predict the effects of design decisions [Jansen and Bosch 2005] on the capabilities, behavior and evolution of a CIS.

In order to provide a consolidated systematic knowledge base of architectural commonalities and variations in the CIS domain, we present in this work five major pattern variants of CIS. We investigated a number of CIS in the field with focus on a detailed survey of existing variants among key architecture-significant principles based on previously identified basic concepts, principles, and characteristics of software architectures that all CIS have in common. The variants are identified along two dimensions with respect to the relationship between key elements across and within two layers of the system. Further the variants build upon and extend the STIGMERGIC INFORMATION SYSTEM (SIS) basis pattern [Musil et al. 2015b] that we reported at EuroPLoP 2015. The SIS architecture pattern enables software architects to describe the core elements and processes of a CIS architecture. Although the SIS pattern describes the commonalities of CIS on a wider scope, it is limited in its capability to describe specifics of CIS subfamilies, such as social networks or wikis. By increasing the knowledge about variability dimensions in CIS, this work provides the foundation for advanced CIS design and architecting approaches, like a pattern language for CIS, CIS domain engineering, and variability-centric reference architectures.

The remainder of the work is structured as follows: Section 2 gives an overview of the SIS architecture pattern. Section 3 describes identified major variants of the SIS pattern. Section 4 presents a discussion and limitations of the results. Finally, section 5 draws conclusions and outlines future work.

2. BACKGROUND: STIGMERGIC INFORMATION SYSTEM PATTERN

In this work we aim to extend the original STIGMERGIC INFORMATION SYSTEM (SIS) architecture pattern that was presented at EuroPLoP 2015 [Musil et al. 2015b] based on emerging pattern variants around the central relations between artifacts, actors that own an artifact, and actors that contribute to an artifact. To introduce the basic concepts and processes in order to understand the pattern variants, we briefly summarize the SIS pattern in this section.

The problem addressed by the SIS pattern is a lack of structured coordination to share and retrieve the knowledge and information between human agents by using a software system [Musil et al. 2015b]. Knowledge and information are dispersed among individuals and thus are difficult to be aware of and access on a collective level. The solution of the SIS pattern provides a minimal system description including the common elements and processes of a CIS. A SIS-based architecture design facilitates bottom-up information sharing and knowledge aggregation by allowing its actors to create/modify user-generated content stored in topic-specific artifacts and enables a CIS to provide self-organizational knowledge transfer and coordination capabilities to human groups and organizations [Musil et al. 2015b].

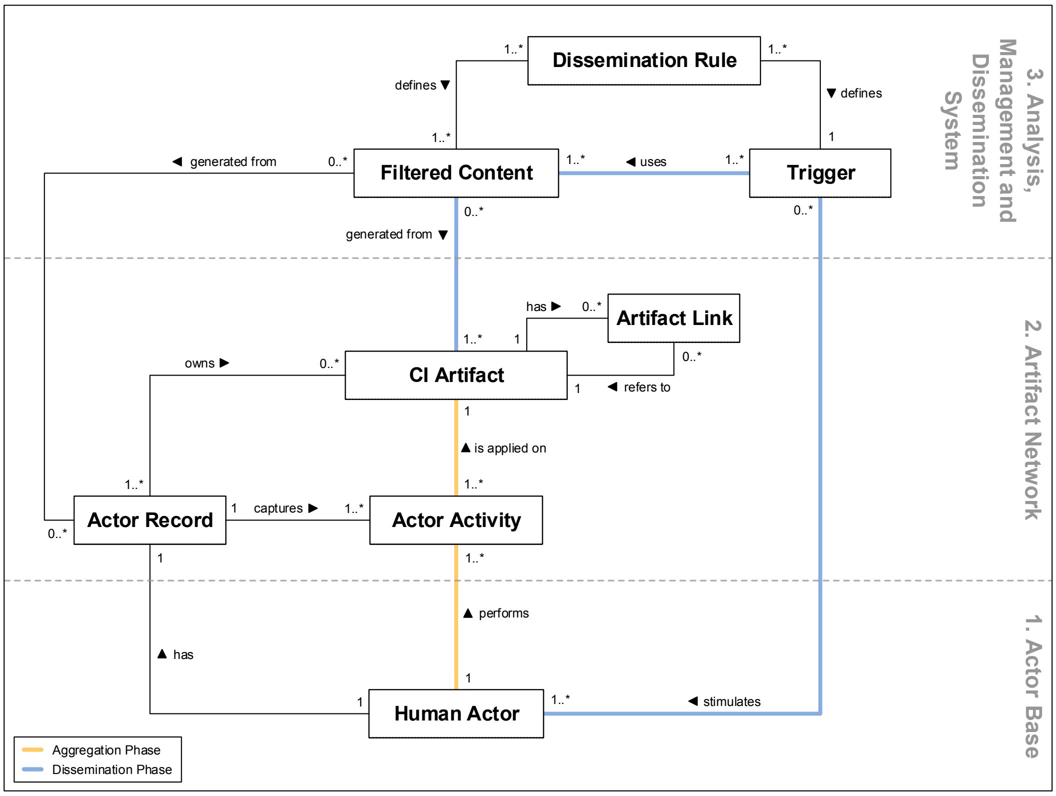

Fig. 1. Metamodel of the original SIS pattern [Musil et al. 2015b] consisting of aggregation (yellow) and dissemination phase (blue).

The SIS pattern [Musil et al. 2015b] describes a system architecture of a hybrid human-computer system that consists of the following three main layers and architectural elements:

1. Actor Base. The actor base as proactive layer consists of *human agents*, who independently interact with the system by performing *activities* on the *CI (Collective Intelligence) artifacts* and thus contribute content.

2. Artifact Network. The single, homogeneous artifact network as a passive layer consists of CI artifacts, which store topic-specific content contributed by the actors. CI artifacts are manipulated by *actor activities*, which resemble different types of create, read, update and delete operations. An important activity is the linking of artifacts using *artifact links* which leads to the emergent creation of an *artifact network*. Each performed activity is tracked in an *actor record* to support the system to provide services like recommendations and shared interests, whereby each actor has her own actor record. In addition, the actor record represents an existing *ownership relation* between the actor and the CI artifacts. The ownership relation defines who is the owner of an artifact and thus has extensive control to decide (1) to which extent other actors are able to contribute to the CI artifact, and (2) if contributions comply to predefined qualitative requirements.

3. Analysis, Management and Dissemination System (AMD). The computational AMD system as a reactive/adaptive layer enforces different *dissemination rules*. Based on these rules, *filtered content* is

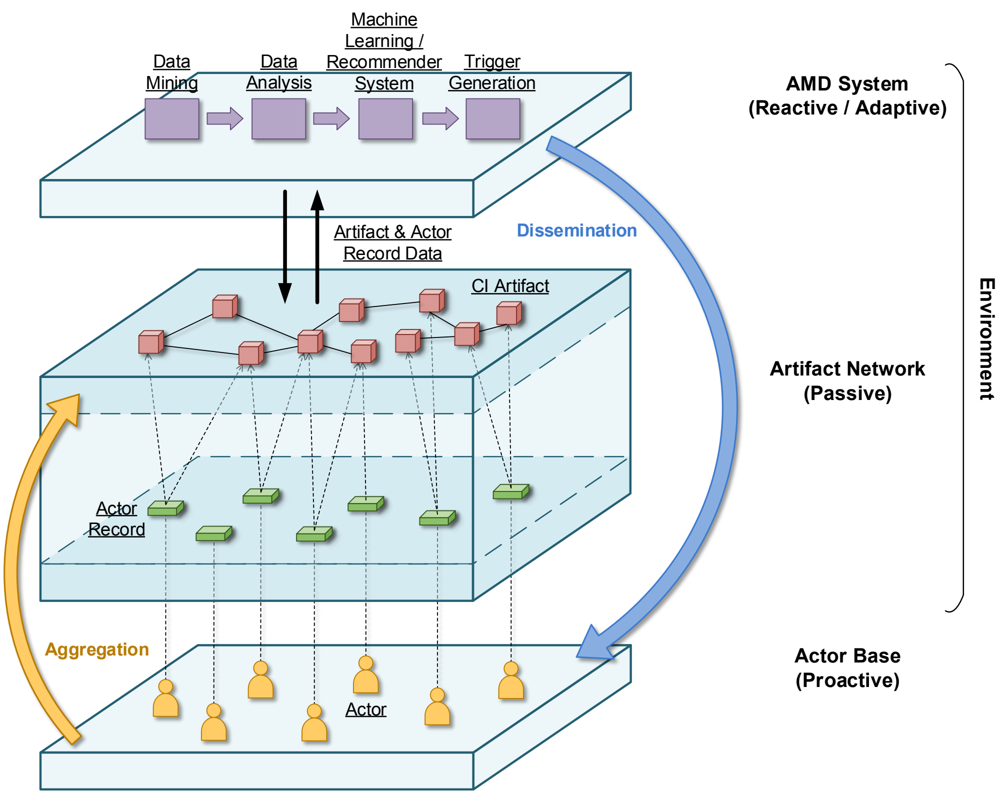

Fig. 2. SIS multi-layer model from Musil et al. [2015a] extended with two sub-layers in the artifact network layer.

generated from CI artifact content and actor records. The filtered content is used by generated *triggers* that are sent to individual actors in order to propagate changes of CI artifacts and increase the awareness about ongoing activities as well as to stimulate subsequent actor activities and artifact contributions, realizing the stigmergic process.

The *stigmergic process* [Heylighen 2016; Ricci et al. 2007] with an aggregation and a dissemination phase is a central process in each CIS. It describes a perpetual, positive feedback loop between the actor base (layer 1) and the coordination infrastructure (layer 2 & 3) that is created by instrumenting the actors' activities and contributions to stimulate subsequent reactions by other actors [Musil et al. 2015b]. The continuous flow of actor contributions within the system environment enables the emergence of collective intelligence that allows the individual to benefit for own purposes. Figure 1 illustrates the metamodel that underlies the SIS pattern and describes core elements and their relations. In addition, it highlights the stigmergic process comprising the aggregation (yellow) and dissemination (blue) phase. For a detailed description of the SIS pattern we refer the interested reader to Musil et al. [2015b]. In the following the metamodel is extended with identified variation points that are characteristic for the presented SIS pattern variants.

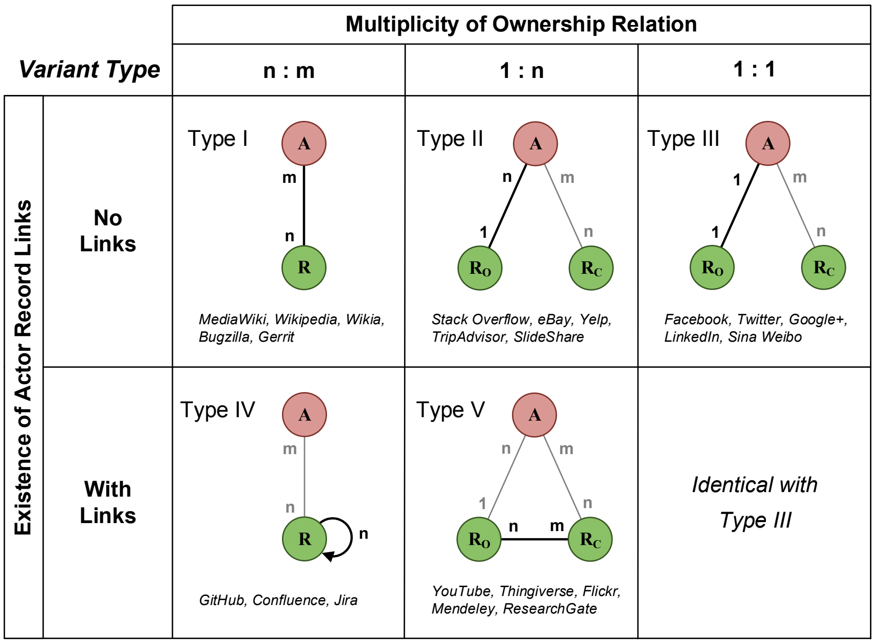

Fig. 3. Overview of the identified variants in the sub-layers of the artifact network layer with example platforms. They are based on the combination of two dimensions: (1) multiplicity of the ownership relation and (2) the existence of actor record links.

3. CHARACTERISTIC VARIANTS OF STIGMERGIC INFORMATION SYSTEM PATTERN

In order to define the variants in the SIS pattern, it is necessary to refine the representation of the original pattern. Figure 2 shows a multi-layer representation of the SIS pattern with the three main layers and the stigmergic process based on Musil et al. [2015a]. But in contrast to the former representation, the artifact network layer is split into two sub-layers: one sub-layer for the actor records and the CI artifact network respectively. The characteristic sub-layer variants (figure 3) are derived from variations in the actor record and CI artifact elements of the base SIS pattern. These variations occur in two dimensions: (1) the ownership relation between the actor record and the CI artifact, and (2) the linking between actor record elements. These variations are architectural significant with respect to a CIS's network topology and thus profoundly impacts how a CIS organizes actors and CI artifacts. In the following, we describe the characteristic SIS pattern variants in more detail. We define the following symbols used in the illustrative figures to explain each variant: **A** represents a CI **A**rtifact, $\mathbf{R_O}$ represents an actor **R**ecord that has an ownership relation to A, and $\mathbf{R_C}$ represents an actor **R**ecord that has a contributor relation to A and thus no ownership relation.

We explain the ownership relation by taking the example of YouTube, where topic-specific information is captured in form of the video. In YouTube, A represents the data structure that encapsulates not only the uploaded video file and its video metadata, but also a name, description, tags, view counter, ratings, comments, privacy settings, list information and platform metadata. An actor who has ownership R_O of a YouTube CI artifact A has complete editorial control of the I content. The actor can replace the video file and change the CI artifact's title, description, privacy settings, tags, but also the access rights if other actors are able to rate and comment the video. The contributing actors are actors who have no ownership relation R_C. Although the actors with R_C are not able to completely modify A, but they are able to enrich A with their ratings, comments, etc. The organization of who has full

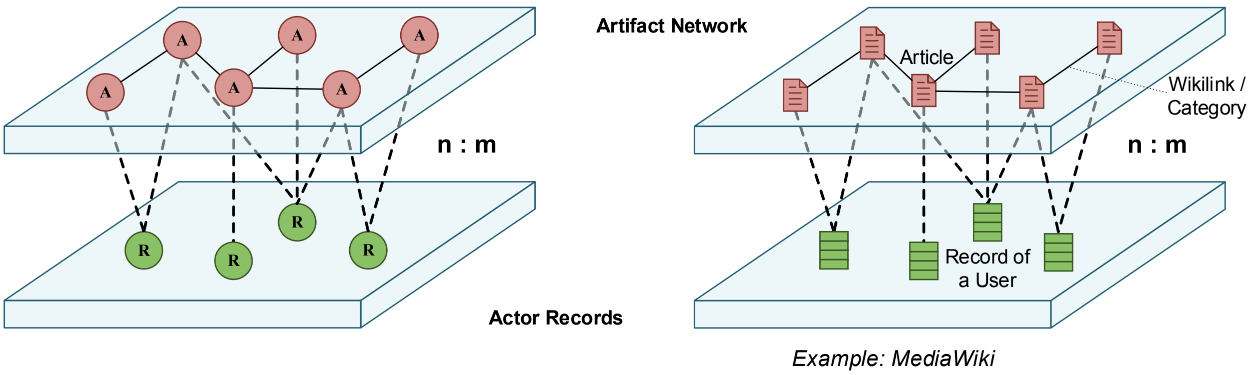

Fig. 4. Type I variant with n:m mapping between actor records (A) and CI artifacts (I), but without existing links between actor records.

editorial control and who can just add / manipulate a subset of the CI artifact's data elements is a leverage point in the design of CIS and thus also needs to be reflected within a more comprehensive SIS pattern.

3.1 *Type I:* n:m-Ownership Relation without Actor Record Links

Type I variant (figure 4) describes a n:m mapping between R_O and A, which means that multiple actors own multiple CI artifacts, whereby each A is owned by more than one R_O. This variant does not support the existence of links between actor records. This pattern variant comprises highly collaborative platforms like wikis or issue trackers. In this variant typically all actors have R_O on all A which gives the actor base more freedom in contributing information, but also requires additional mechanisms for content quality assurance and vandalism protection. Type I platforms are particularly popular in corporate intra-organizational settings, since in this setting actors (specifically employees) effectively can be held accountable for grave editorial violations and malicious misconduct.

Example: **MediaWiki[1].** The CI artifact in MediaWiki is the article (A). All actors (R_O) can create new articles and modify the full content of existing ones. Articles can be linked together by users via Wikilinks and categories, creating a network of related articles.

Further platform examples: Wikipedia[1], Wikia[2], Bugzilla[1], Gerrit[3].

3.2 *Type II:* 1:n-Ownership Relation without Actor Record Links

Type II variant (figure 5) describes a 1:n mapping between R_O and A, which means that each actor owns multiple CI artifacts and there is an indefinite number of R_C for any A. This variant does not support the existence of links between actor records. This pattern variant comprises platforms which include a majority of social platforms. Typically A realizes a concept, where a certain content baseline is defined by the owner, but then requires an actor base of R_C to extend certain areas of A, for example with ratings, comments or reviews.

[1] *http://www.{name}.org* (all URLs last accessed: 06/12/2016)
[2] *http://www.{name}.com* (all URLs last accessed: 06/12/2016
[3] http://www.gerritcodereview.com (last accessed: 06/12/2016)

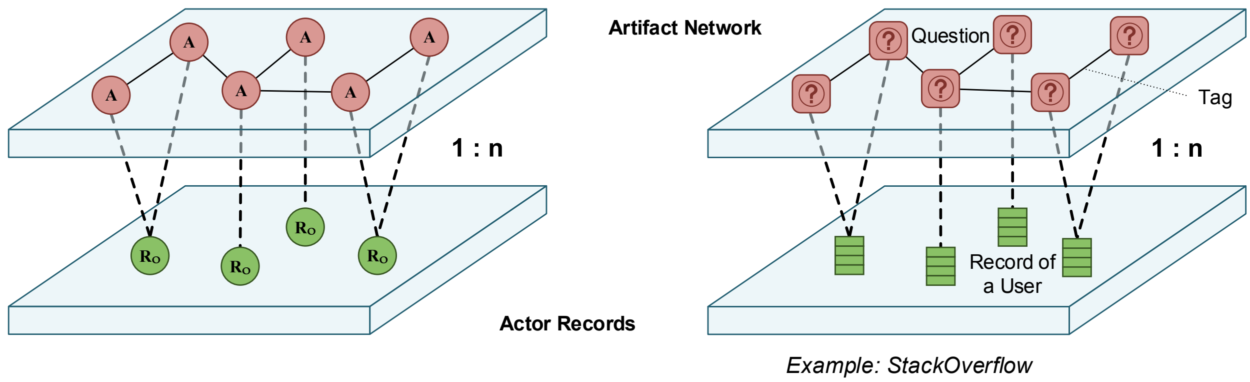

Fig. 5. Type II variant with 1:n mapping between actor records (R_O) and CI artifacts (A), but without existing links between actor records.

Example: **Stack Overflow[2].** The CI artifact in Stack Overflow is the user question. Each actor can be the owner (R_O) of several questions. Other actors (R_C) can answer the questions, but also upvote or downvote the answers of other actors with respect to their correctness and usefulness. Artifact links are represented by tags that actors can assign to questions.

Further platform examples: eBay[2], TripAdvisor[2], Yelp[2], SlideShare[4], Pinterest[2], Zillow[2], Urban Dictionary[2], CrunchBase[2].

3.3 *Type III:* 1:1-Ownership Relation without Actor Record Links

Type III variant (figure 6) describes a 1:1 mapping between R_O and A, which means that each actor only owns one CI artifact, but there is an indefinite number of R_C for any A. This variant does not support the existence of links between actor records. This pattern variant comprises platforms which are known as social networking services. Typically A realizes a concept which is closely releated to the

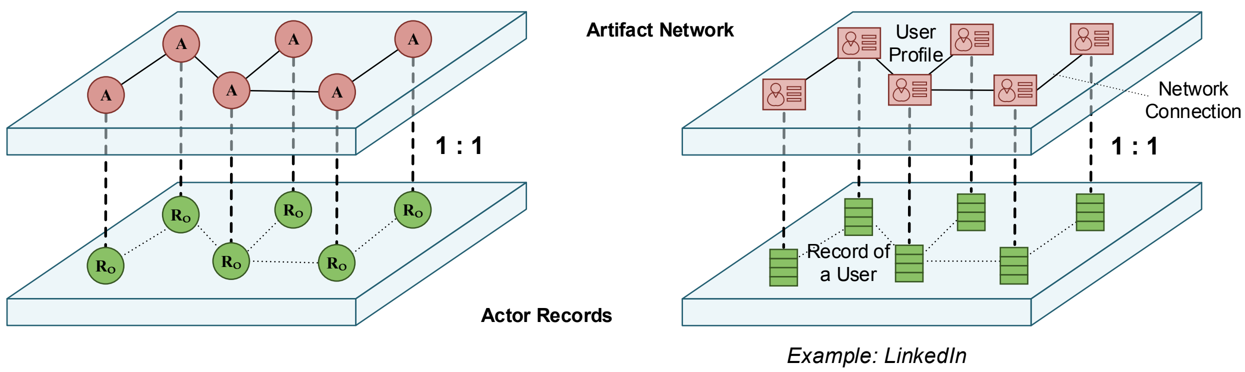

Fig. 6. Type III variant with 1:1 mapping between actor records (R_O) and CI artifacts (A), but without existing links between actor records.

[4]http://www.slideshare.net (last accessed: 6/12/2016)

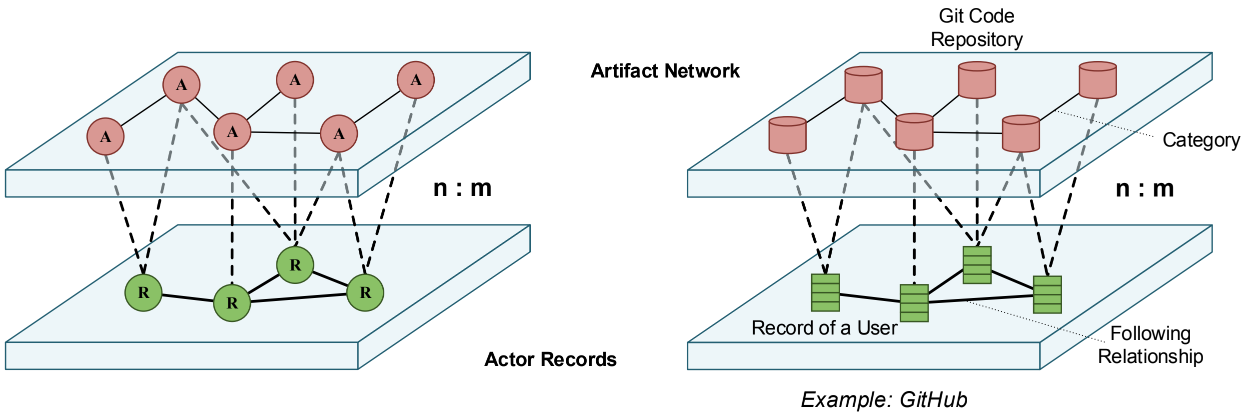

Fig. 7. Type IV variant with n:m mapping between actor records (R) and CI artifacts (A) and with additional social network on the actor record sub-layer.

actor, for example a profile or a set of content, that is exclusively produced by R_O.

Example: **LinkedIn[2].** The CI artifact in LinkedIn is a user's professional profile (A). Each actor is the owner (R_O) of one single profile. Actors primarily contribute to their own profile pages, but they can also contribute (R_C) directly to information shared by other users or their profile pages. Artifact links are represented by the professional network connection between two profiles and are defined by the actors.

Further platform examples: Facebook[2], Twitter[2], Google+[5], Instagram[2], Xing[2], ResearchGate[2].

3.4 *Type IV:* n:m-Ownership Relation with Actor Record Links

Type IV variant (figure 7) is identical to type I except that actors are also able to create links between their actor records. In this pattern variant highly collaborative platforms like wikis or social code repositories are extended with an additional social network on the actor record sub-layer.

Example: **GitHub[2].** The CI artifact in GitHub is the Git code repository (A). All actors (R_O) can create new repositories, upvote and can modify the content of existing repositories. The integration of changes on repository level follows the procedures of the Git versioning system. Repositories are linked via categories. Actors can follow each other, and so create a social network on actor record level.

Further platform examples: Confluence[6], Jira[7].

3.5 *Type V:* 1:n-Ownership Relation with Actor Record Links

Type V variant (figure 8) is identical to type II except that actors are also able to create links between their actor records. This pattern variant extends a social platforms with an additional social network on the actor record sub-layer. Often type V variants start initially as type II and later get functionality

[5]http://plus.google.com (last accessed: 6/12/2016)
[6]http://www.atlassian.com/software/confluence (last accessed: 6/12/2016)
[7]http://www.atlassian.com/software/jira (last accessed: 6/12/2016)

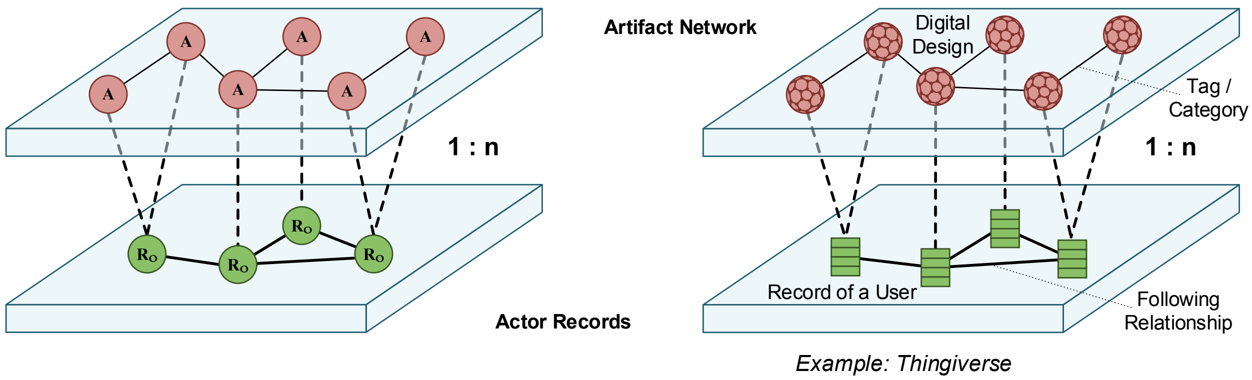

Fig. 8. Type V variant with 1:n mapping between actor records (R_O) and CI artifacts (A) and with additional social network on the actor record sub-layer.

added to befriend, subscribe or follow a user.

Example: **YouTube[2].** The CI artifact in YouTube is an uploaded video (A). Each actor can be the owner (R_O) of multiple videos, that she uploaded. The owner is allowed to define tags and categories of the artifact which represents the artifact links. Other actors (R_C) can watch and typically rate and comment the videos as well as subscribe to channels (lists of videos). Also actors can subscribe to other actors, which generates a social network on actor record level.

Thingiverse[2]. The CI artifact in Thingiverse is a digital design (A) which can be printed by a 3D printer. Each actor can be the owner (R_O) of several shared 3D designs. Actors can assign tags/categories to 3D model designs, but they can also add artifacts to collections, which generates different kinds of artifact links. Other actors (R_C) can comment, upvote and remix designs, and organize them in collections. Actors can also follow each other, and so create a social network on actor record level.

Further platform examples: Flickr[2], Mendeley[2], Foursquare[2].

4. DISCUSSION

The SIS pattern provides a universal model of CIS, whereby this work explains differences between systems with variants in sub-layers of the original pattern. The identified variants present a refinement of the SIS pattern with regard to the degree of dependence of a CI artifact on an individual actor. Figure 9 illustrates the adapted metamodel of the SIS pattern that is extended with the identified variation points: existence of links between actor records, the multiplicity of the ownership relation to a CI artifact, and a differentiation between activities that can be performed only by the actor who is owner of a CI artifact and activities for contributions of other actors.

CI artifacts in the type III variant (1:1-ownership relation without actor record links) are fully dependent on a particular actor, for example a profile on Facebook or Twitter. In constrast CI artifacts in the type I and IV variants (n:m-ownership relation without/with actor record links) are less dependant on a specific actor, like in a Wiki system.

A particular finding in the type III variant scenario is that the topology of the CI artifact and actor record network sub-layers are identical in both variants of the actor record sub-layer, thus creating

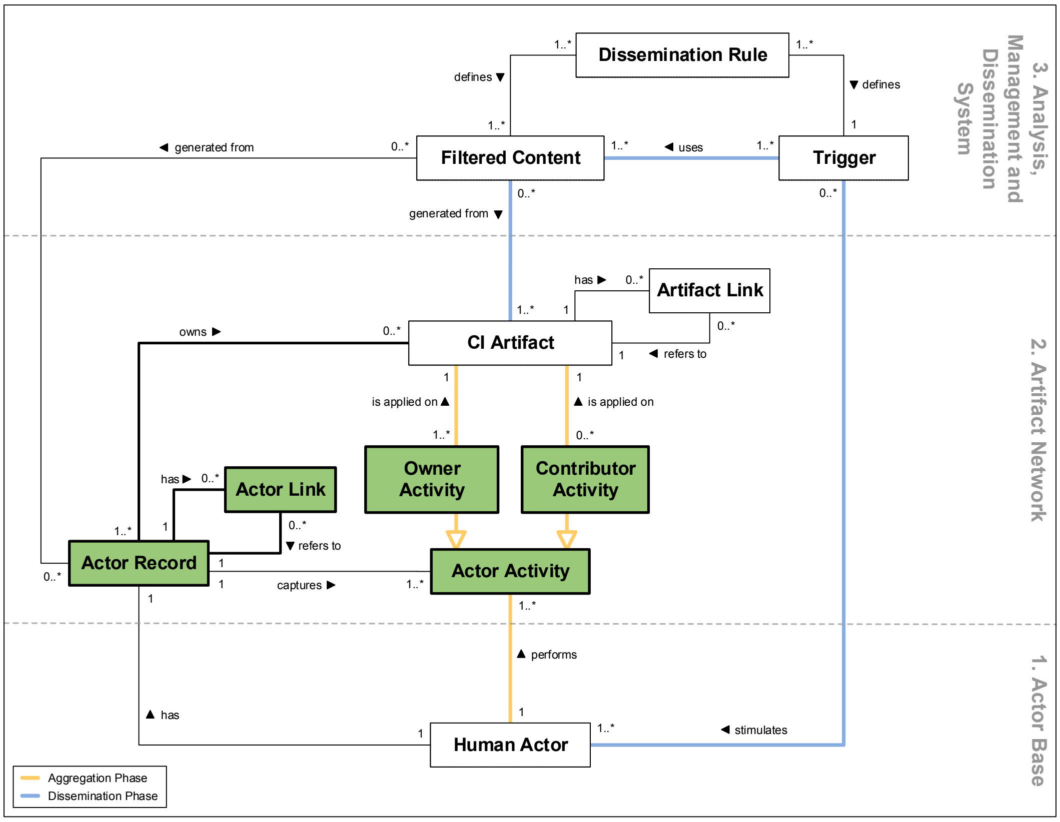

Fig. 9. Metamodel of the extended SIS pattern highlighting the variation points that represent the basis of the pattern variants (refined from Musil et al. [2015b]).

an identical graph. This finding provides implications for future evaluations of CIS, since different sub-layer variants may also require adaptation in network analysis methods.

Although the presented variants are the result of a detailed investigation of a number of CIS in the field, it takes an extensive and systematic review of CIS in a wider scope to conclusively validate the identified architecture-significant variants and to investigate further variation points so that more patterns can be derived leading to a large CIS family description. Despite increasing the degree of detail of the original SIS pattern by the presented variants, an investigation of additional approaches for software architects is essential to better support the design and architectural decision making processes of CIS, like architecture frameworks or architecture description languages. These approaches need also to address these CIS variants, guide the design of them and evaluate trade-offs between the different variants as well as the options for the variation points based on the intended purpose and business goals of the system before making important design decisions. Furthermore, it is important to note that the presented pattern variants are conceptual patterns that can be technically implemented in different ways depending on what other technical paradigms (like service-oriented architecture or micro-services) are used.

5. CONCLUSION

In this work, we extended the STIGMERGIC INFORMATION SYSTEM (SIS) pattern with five major pattern variants which are composed of two architecture significant dimensions: (1) the multiplicity of the ownership relation between the actor record and the CI artifact, and (2) the existence of links between actor record elements. These variants show that some kind of grouping is emerging around the original SIS pattern and thus are useful to describe larger CIS families. The results of this work provide the foundation of advanced CIS architecting approaches like an architecture pattern language for CIS or a reference architecture for one of the identified types. Future research also needs to validate the presented variants with an extensive systematic review of CIS and to individually investigate each of the identified types in order to identify further characteristics and patterns.

ACKNOWLEDGMENT

We thank our shepherd Dietmar Schuetz and the participants of the writers' workshop group at EuroPLoP 2016 as well as Danny Weyns for their constructive and helpful feedback and comments to improve the paper.

REFERENCES

Luis Von Ahn. 2005. *Human Computation*. Ph.D. Dissertation. Carnegie Mellon University, Pittsburgh. http://ra.adm.cs.cmu.edu/anon/usr/ftp/2005/CMU-CS-05-193.pdf

Tom Gruber. 2008. Collective Knowledge Systems: Where the Social Web meets the Semantic Web. *Semantic Web and Web 2.0* 6, 1 (2008), 4–13.

Francis Heylighen. 2016. Stigmergy as a universal coordination mechanism I: Definition and components. *Cognitive Systems Research* 38 (2016), 4–13.

Jeff Howe. 2006. The Rise of Crowdsourcing. *Wired* 14, 6 (jun 2006). http://www.wired.com/wired/archive/14.06/crowds.html

Anton Jansen and Jan Bosch. 2005. Software Architecture as a Set of Architectural Design Decisions. In *Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture (WICSA '05)*. IEEE, 109–120.

Daniele Miorandi, Vincenzo Maltese, Michael Rovatsos, Anton Nijholt, and James Stewart (Eds.). 2014. *Social Collective Intelligence: Combining the Powers of Humans and Machines to Build a Smarter Society*. Springer International Publishing.

Juergen Musil, Angelika Musil, and Stefan Biffl. 2015a. Introduction and Challenges of Environment Architectures for Collective Intelligence Systems. In *Agent Environments for Multi-Agent Systems IV*, Danny Weyns and Fabien Michel (Eds.). LNCS, Vol. 9068. Springer International Publishing, 76–94.

Juergen Musil, Angelika Musil, and Stefan Biffl. 2015b. SIS: An Architecture Pattern for Collective Intelligence Systems. In *Proceedings of the 20th European Conference on Pattern Languages of Programs (EuroPLoP '15)*. ACM, 20:1–20:12.

Juergen Musil, Angelika Musil, Danny Weyns, and Stefan Biffl. 2015c. An Architecture Framework for Collective Intelligence Systems. In *Proceedings of the 12th Working IEEE/IFIP Conference on Software Architecture (WICSA'15)*. IEEE, 21–30.

Andrea Omicini and Pierluigi Contucci. 2013. Complexity and Interaction: Blurring Borders between Physical, Computational, and Social Systems. Preliminary Notes. In *Proceedings of the 5th International Conference on Computational Collective Intelligence Technologies and Applications (ICCCI '13) (LNCS)*, Costin Badica, Ngoc Thanh Nguyen, and Marius Brezovan (Eds.), Vol. 8083. Springer Berlin Heidelberg, 1–10.

Manoj Parameswaran and Andrew B. Whinston. 2007. Social Computing: An Overview. *Communications of the Association for Information Systems* 19, 37 (2007), 762–780.

Alessandro Ricci, Andrea Omicini, Mirko Viroli, Luca Gardelli, and Enrico Oliva. 2007. Cognitive Stigmergy: Towards a Framework Based on Agents and Artifacts. In *Proceedings of the 3rd International Conference on Environments for Multi-Agent Systems III (E4MAS '06) (LNCS)*, Danny Weyns, H. Van Dyke Parunak, and Fabien Michel (Eds.), Vol. 4389. Springer Berlin Heidelberg, 124–140.

Elena Simperl. 2015. How to Use Crowdsourcing Effectively: Guidelines and Examples. *LIBER Quarterly* 25, 1 (2015), 18–39.

Danny Weyns, Andrea Omicini, and James Odell. 2007. Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems* 14, 1 (2007), 5–30.