Strategies for Harnessing the Collective Intelligence of Cultural Institutions' Communities

Considerations on Supporting Heterogeneous Groups in Content Production Taking the Quality Factor into Consideration

Leonardo Moura de Araújo
dimeb – Digitale Medien in der Bildung, University of Bremen, Bremen, Germany
araujo@informatik.uni-bremen.de

Keywords: Content Production, Platforms, Collective Intelligence, Cultural Institutions, Scaffolding, Design Thinking.

Abstract: This support paper builds a theoretical model upon which computer platforms for cultural institutions can be based upon. It analyses three landmark models in the Computer Science history that were capable of harnessing the Collective Intelligence present on gravitating communities. Afterwards, conclusions are drawn regarding their effectiveness in leveraging communities. Instructional Scaffolding and Design Thinking are indicated as important strategies to provide the necessary support to heterogeneous groups.

1 INTRODUCTION

Due to the increase in popularity of social media services and strategies across digital and physical landscapes, there is growing expectation and pressure for Cultural Institutions (CIs) to make the transition from static providers of cultural content to flexible facilitators of interaction, participation, and collaboration among their audiences.

In order to meet those expectations e.g. some museums started to incorporate 2.0 philosophies into their venues to make their role more meaningful to their communities. From interactive installations to collective creation and remixes of content, those initiatives represent an effort of museums to bring students, professionals, hobbyists, aficionados, and basically anyone to be part of the making process.

The outcomes of participatory attitudes that see the public as creative agents can be considerably significant. On one hand, CIs profit from the content and knowledge produced by their audiences. On the other hand, the public perceives CIs as compelling spaces where they can express themselves.

One of the challenges in giving the public the chance to express themselves regards the quality of the content they produce using technology. Poor outcomes can be originated not only by the lack of clarity, clear instructions, and support from the part of the museum app, but also individual deficiencies such as limited experience and knowledge from the part of the public. How can CIs support their public in producing high-quality and reliable content with technology?

This position paper examines successful models in the Computer Science (CS) field that were capable of harnessing the Collective Intelligence to produce high-quality outcomes. They are: the open-source model (OSM), collaboration applications (CA), and software platforms (SP).

Those models offer hints of how to think collaborative apps that take advantage of gravitating communities effectively. However, apart from CAs such as Wikipedia, those models are directed at professionals from the CS field, and do not focus on individuals with little or no formal training. Even Wikipedia can be harsh with inexperienced users.

As a solution for attempting to support inexperienced collaborators of CIs, this paper points out to two well-known methodologies, namely Design Thinking (DT) and Instructional Scaffolding, that are capable of facilitating and supporting the production of content by large and heterogeneous groups when integrated in computer applications.

2 THE COLLECTIVE INTELLIGENCE

Due to new communication technologies, individuals collaborate in a diversity of ways never possible before. On the Internet, the Collective Intelligence is manifested not only by single and collaborative works built upon previous knowledge, but also in the human-human and human-computer relationships on the network.

The vision of past about human-computer symbiosis turned out to be to a certain extent revealing. In 1960, Licklider (Licklider, 1960) in his article *Man-Computer Symbiosis* predicts that human brains and computers would be coupled tightly together and that "the resulting partnership will think as no human brain has ever thought and process data in a way not approached by the information-handling machines we know today" (Licklider, 1960).

Man-made networks connect millions of individuals, who consist of different kinds of biological networks themselves. Neural networks of the human brain share similarities with the Internet regarding their structure and functionality. Eguiluz et. al. describe super connected brain regions in neural networks that appear to work as "hubs" facilitating the communications between distant "nodes" or specific and less connected brain areas [see (Eguiluz et al., 2005)]. Those "hubs" operate in a comparable way to search engines in that they do not hold the information, but point to the location to where it is.

Macro and micro resemblances might hold answers why we are so comfortable with computers. This configuration between biology and technology, the "Global Brain", holds the potential "to become truly transformative in domains from education and industry to government and the arts" (Bernstein et al., 2012).

2.1 Landmark Models

The *Global Brain* and its collective intelligence foundations, which are the result of human-computer symbioses, can be better examined by looking at landmark developments of the computer history such as the OSM, CA, and SP architectures.

2.1.1 The Open-source Development Model

Open-source is perhaps one of the most well known examples of collaborative work that uses collective intelligence to produce high quality outcomes. Its development model is focused on both the premise of users being considered as co-developers and free-license agreements.
Open-source projects are probably ideal examples of almost non-hierarchical collaborative production processes that are able to cope with highly complicated and large amount of data. As Weber (Weber, 2000) shows, those collaborative processes raise questions regarding motivation of contributors, coordination of projects, and complexity of communities.

In most cases, the OSM does not work on the premise of monetary rewards, but appeals to people's individual motivations, such as having fun with programming puzzles, contributing to projects that are socially relevant, gaining visibility regarding programming skills, and profiting from more experienced individuals [see (Weber, 2000)].

The Internet stands as a facilitating tool able to deal with large numbers of collaborators. On one hand, the virtual space the Internet provides is capable to scale in a way that physical spaces are not. Almost "unlimited" space can be allocated for communication, data storage, working branches, and individual and collective profiles. The stored data can be picked up and worked further at anytime, anywhere. Therefore, having enough space for specific code branches is important to differentiate and keep projects organized. On the other hand, intelligent algorithms created for e.g. revision control aid programmers to keep track of changes on the code, and help to identify individuals responsible for those modifications.

The distributed configuration of open source projects presents challenges regarding management and control of the work produced. Solving conflicts within groups is also a concern that should be taken into consideration. In the case of Linux, Linus Torvalds is seen as an authority in that he was the founder of the Linux Kernel project. In this sense, Torvalds and programmers who are high in the hierarchy have the last word in decisons. Apart from that, and in most cases, the code decides.

2.1.2 Software Platforms

In Product and System Design, a platform is a structuring foundation on top of which a set of independent elements and interfaces can be arranged, rearranged, and innovated upon [see (Griffiths, 2010)]. Shared key components and assets define the core of the platform and diversification can be achieved by building upon and extending capabilities to build new, but related foundations. Baldwin et. al. (Baldwin and Woodard, 2009) point out that most platform definitions identify the reuse or sharing of common elements as core characteristics, and that all platforms are "modularizations of complex systems in which certain components (the platform itself) remain stable, while others (the complements) are encouraged to vary in cross-section or over time“ (Baldwin and Woodard, 2009). Well-structured platforms allow numerous advantages, such as cost saving, increased production efficiency, ability to evolve and produce variety in large scale.

In CS, this term was initially used to define the computing hardware and later on the operating system (OS) upon which programs would run. Earlier computers had to be built from the ground up always when new releases were planned. Not infrequently, because of incompatibility with newer systems, there was a costly process involved in moving data to different formats.

As computer hardware was becoming modular and increasing its complexity, systems without OSs presented enormous challenges. Earlier computers required the full hardware specification to be attached to the application every time it ran. Therefore, a program not only would be suitable for just one machine, but also it had to be loaded every time someone needed the program. As a way to optimize computers, the industry realized that some fundamental set of instructions could be loaded into the memory and managed separately. The distinction between applications and OS was then created.

The reuse of code and the modularization of computer systems brought many advantages to the field especially regarding cost, manageability, and evolution of complex systems. According to Baldwin et. al., “by promoting the reuse of core components, such partitioning can reduce the cost of variety and innovation at the system level. The whole system does not have to be invented or rebuilt from scratch to generate a new product, accommodate heterogeneous tastes, or respond to change in the external environment“ (Baldwin and Woodard, 2009).

The modularity of computer systems became even more robust with the advent of Object-Oriented Programming (OOP) that took the reusability of software components to a next level. Powerful software development frameworks reduced the complexity and cost of writing code. Frameworks “mean a real breakthrough in software reusability: not only single building blocks but whole software (sub-)systems including their design can be reused.”(Pree, n.d.)

The inexpensive and powerful strategies allowed by frameworks create a rich ecosystem for application development that lead to an enormous variety and innovation, especially when it is popular and open to a great number of developers.

2.1.3 Collaborative System Model

Collaborative systems are designed to support individuals to accomplish tasks in a cooperative manner. The free-content encyclopedia Wikipedia is a successful example of a tool based on an openly editable model that came from the open source experience, which sees users as contributors. Wikipedia covers an enormous amount of subjects. More than 22,000,000 articles were written so far. As in open source software projects, the outcome of the collaborative process lead to the creation of articles “of remarkably high quality“ (Malone et al., 2010, p. 21).

Originally, Wikipedia was proposed as a complementary project for its predecessor Nupedia as a mean to generate faster and larger amounts of content. Although also thought to be collaborative, Nupedia was not able to release consistent amount of articles during its existence. The main problems Nupedia had were its highly bureaucratic publication procedure, elitist selection of contributors, and non-wiki platform. Nupedia’s content approval process had seven complicated steps. Non-expert individuals willing to contribute would most likely be vetted with only few cases of exception.

Therefore, Jimmy Wales and Larry Sanger decided to run Wikipedia to facilitate the productions of articles that later would go for the Nupedia’s reviewing process. Basically, the idea was to find a way that uncredentialed people could participate more easily. The differential of Wikipedia relied on its deep open-source philosophies, which considers everything as a “draft in progress, open to revision” (Rettberg, 2005). Impressively, the quality of the articles and the fact the Wikipedia is not prone to amateurism and vandalism is due to the community that is “passionate about the topics they know and care about”(Rettberg, 2005).

The wiki architecture is key to the success of Wikipedia. Anyone is able to contribute including anonymous users. Wikipedia allows contributors to publish their unfinished drafts, so others can make improvements on them. However, the decision of either accepting or not unfinished contributions depends on the patrol division which checks new articles shortly after they are created. Sometimes arbitrary decisions can be made and contributors should engage in discussions to clarify uncertainties. This process can be very localized and specific to the people in charge of approval. In any case, individuals are driven to cooperate if not with writing new publications, then with linking keywords among articles, fixing spelling mistakes, or improving clarity of sentences.

Nothing else holds Wikipedia from being just a wiki than its policies and guidelines. The content created is free for others to read and modify. This is a strong motivation for people to work in something they believe to be valuable for the good of the society. However, Wikipedia has a different culture from regular wikis, because "it's pretty singlemindedly aimed at creating an encyclopedia." (Sanger, n.d.). Although the architecture of wiki software encourages openness and de-centralization allowing deviant content being fed into the system, the community of wikipedians is compliant with the Wikipedia's five fundamental pillars [see ("Wikipedia:Five pillars - Wikipedia, the free encyclopedia," n.d.)].

2.2 The Building Blocks for Quality

What are the building blocks from those models that can be applied to the design of systems capable of harnessing communities' collective intelligence to generate new content with acceptable quality?

In the case of the OSM and CA, the community is the force behind their success, because they see users as co-developers and as part of the decisions. In the case of the open-source, the easy access to the code along with the licenses is one of the most important aspects. Each member of the community has unique needs that are met without them having to reinvent the wheel. This is a direct benefit. At the same time, new extensions, adaptations, and improvements automatically appear in the process. In the open-source movement, the product comes from improving and building on other people's knowledge. And because there is no clear hierarchical authority, in most cases, the quality of the code decides whether it will be popular.

Other motivational aspects are important in leveraging the community. Individual motivations such as contributing with projects that are fun, challenging, educationally beneficial, socially relevant, and can improve one's reputation play a big role in the community. As for the Wikipedia, however, the natural selection that takes place in the case of open-source projects does not happen in the same way. That is because not only the goal of this model is different, but also Wikipedia is more centralized and hierarchically structured. Wikipedia architecture does invite users to create new articles and modify existing ones, and doing so is easy to the extent of not needing to create a user account on the website in order to contribute. The facilitated process of contribution generates an enormous quantity of new content.

Differently from code, that needs to be logically coherent in order to compile, textual information is subtle in hiding factual inaccuracies and non-neutral point of views. Besides having good structure and style, Wikipedia articles need to gather support from other textual information, which takes the form of references, working in a similar way as in academic texts. In order to check for inconsistencies and errors, Wikipedia relies deeply in few contributors, such as the administrators and patrollers. Because of its textual, localized, and more hierarchical structure, it allows certain contradictions inside the system. Although Wikipedia accepts contributions from anonymous users, it is up to the patroller e.g. to accept or not to engage in a discussion about an article that is about to be deleted depending on whether or not the contributor has a user account. Another example is that a contributor would be willing to start a topic with a small contribution and expect others to build on that. But once again, the patroller can expect that a certain threshold to be crossed regarding the amount of text in the article, no matter if Wikipedia in fact incentives small contributions. In any case, although frustrating for newbies, the Wikipedia process is able to generate high-quality content that comes primarily from the large amount content being produced everyday together with its hierarchical selection.

In the case of SPs, its power is in its modularized structure, stable layer of components, and the possibility of numerous configurations provided by a set of compatible and independent elements. The quality factor from platforms comes from the easy and inexpensive experimentations that can be produced. In all three examples, quantity seems to be a determinative factor for quality. One of the cores of the OSM is the fact that it is also based on SPs and modularization is one of the aspects that allow the reuse of code, which leads to cheap and countless forked versions of software.

3 SCAFFOLDING AND DESIGN THINKING

When thinking about harnessing the collective intelligence of the public of CIs for producing content, one should be careful to be as inclusive as possible regarding the public. The three models presented above offer many hints of how to leverage the crowd. One of the most important of them is undoubtedly seeing the members of the community as active co-creators. But the primary focus of those models is on the production of digital content. Supporting and educating users is something that can or not come along during the production process in those models. Acquiring skills to write computer programs or articles is usually seen as pre-requirement in order to be a respected contributor of those movements. CIs, on the other hand, have as one of their primary focus educating their public. Therefore, educating while producing content is a desirable configuration for CIs.

The OSM is perhaps the model that offers the most extensive support for its users. The GNU project defines open-source software as a matter of individual liberty and a right for the user "to run, copy, distribute, study, change and improve the software" ("The GNU Operating System," n.d.). In this sense, educational aspects are considered when talking into account that the user has the right to analyze and study how a program works and how the code should be written. The community plays also an important role regarding educating new generations of programmers in that there is a lot of knowledge exchange in forums.

Open-source software usually goes along with a strong community that provides extensive support. In addition to that, because open-source is highly decentralized, the need for consensus is not a constant issue, as in the case of Wikipedia. That means that the OSM gives more space for individual contributions. That allows small groups of people with the same interests to take further the development of specific code that might not be of interest to a large community.

The wiki architecture, on the other hand, searches for consensus and does not give space for beta versions of articles, what might be a good strategy for supporting beginners. The higher the degree of modularity, the easier it is for contributing. This strategy is deeply used in programming languages and SPs, but no good solution was thought yet regarding text. The history strategy used on Wiki articles fragments information by tracking changes and creating small chunks containing the modifications. Nevertheless the larger the article, the more difficult it is to manage those changes.

In order to make the creation of content highly inclusive, it is not only necessary to modularize and give space to advanced users and beginners, but also to ensure that they get proper support. Moreover, the community should also be open with dealing with poor contributions not by prompting them for exclusion, but driving the contributors towards refinement. Instructional scaffolding therefore is a good strategy in this regard.

The concept of scaffolding was introduced by Wood, Bruner, and Ross [see (Wood et al., 1976)] but derives from the socio-constructivist theories of Vygotsky on the "zone of proximal development". Scaffolding is the assistance provided by experienced individuals that enable inexperienced ones to succeed in tasks that otherwise would be too difficult. Nowadays, this concept has been changed and adapted taking into account computer-based learning environments. Some scaffolding strategies are automatically integrated in software, making it unnecessary e.g. the presence of human assistance. Those strategies can including measures that "induce and stimulate cognitive, metacognitive, motivational, and/or cooperative activities during learning" (Raes et al., 2012). Scaffolding can also support ways for individuals to keep track of overall plans and progress, which are common obstacles faced in tasks that require learning. One way of doing so is automatically handling nonsalient and routine tasks, reducing the cognitive load while executing a task. This allows learners to focus only on what is important.

As for CIs, Nina Simon (Simon, 2010) talks about the importance of instructional scaffolding when museums ask visitors to create content. One of the problems, is that "open-ended self-expression requires self-directed creativity" (Simon, 2010). That can be difficult especially for would-be participants since it leads to the situation where "participants have to have an idea of what they'd like to say or make, and then they have to produce it in a way that satisfies their standards of quality" (Simon, 2010). By directing, scaffolding limits the participation, but allows visitors to feel more confident and confortable in engaging with CIs.

While instructional scaffolding is able to provide necessary support, DT principles can be used to drive individuals to advance their knowledge in creative ways. As Welsh et. al. (Welsh and Dehler, 2012) point out, "design thinking's emphasis is on developing possibilities rather than satisfying constraints" (Welsh and Dehler, 2012). DT counts on abilities such as intuition, pattern recognition, and generation of ideas that are emotionally meaningful. Furthermore, it eliminates the fear of failure, because it is experimental and non-judgmental. It is human-centered, meaning that it considers the point of view of the community. It stimulates great amount of input that is allowed by thinking *out of the box*. In the context of CIs those features are desirable, because they are inclusive, especially when taking children into account. The most important contributions of DT in the context of CIs are methods for defining important issues through empathy and gathering insights in a non-critical way, that can be later explored.

It can be hard to conciliate a methodology bounded for innovation and experimentation with instructional scaffolding that tries to limit open-ended possibilities. Notwithstanding the contradictions, DT can be seen in fact as a scaffolding strategy for leading contributors towards specific goals. And, because of its flexible nature, it can be adapted to particular contexts regarding its phases, and methods. From more strict approach, when dealing to historical facts, to more associative thinking, when trying to interpret an abstract painting, those two methodologies combined hold the potential to help the crowd to make sense and produce original content based on their own interpretation and research.

4 CONCLUSIONS

Contributing to the construction of knowledge, and being inclusive and innovative concerning the outcomes produced by the public of CIs is no easy task. Creating content for CIs means dealing with open-ended possibilities, heterogeneous groups, and require significant disciplinary knowledge and metacognitive skills. Those requirements are not always met. In this sense, technology appears not only as an instrument for inclusion, but also as a facilitating tool to accomplish tasks.

In the CS field, the OSM, CAs, and SPs are three landmark models for harnessing the collective intelligence of gravitating communities to produce reliable and high-quality outcomes. Some of their most important features are:

- Dealing openly and fairly with the community they serve to.
- Individuals are seen as co-creators and have their say regarding the direction and shape of most open-source projects and CAs.
- Free access to the core is seen as a right concerning open-source philosophies. This is guaranteed by comprehensive licenses.
- Building on other people's knowledge is supported by free access to code, text, and structures those models offer.
- Modularization is a key factor for producing derivations, because it reduces costs by promoting reuse of elements.
- The capability of producing great amount of outcomes lead to increase quality.

Those models however are focused on the production of digital content. Support and education of users are in many cases not considered. CIs, on the other hand, have as one of their primary goals informing and educating their public. Therefore, it is a desirable configuration to include sense making and knowledge building into the design of platforms for CIs. Instructional scaffolding and DT offer valuable hints in this regard. They are able to organize the creative process, directing it towards innovation and experimentation by providing the necessary building blocks in the same way platforms do. Scaffolding promotes focus. DT offers human-centered strategies for empathy, need finding, and generation of insights in a non-critical way.

REFERENCES

Baldwin, C. Y., Woodard, C. J., 2009. The architecture of platforms: A unified view. Platf. Mark. Innov. 19–44.

Bernstein, A., Klein, M., Malone, T. W., 2012. Programming the Global Brain. Commun. ACM May 2012 Vol 55 Issue 5.

Eguiluz, V. M., Chialvo, D. R., Cecchi, G. A., Baliki, M., Apkarian, A.V., 2005. Scale-free brain functional networks. Phys. Rev. Lett. 94, 018102.

Griffiths, D., Phillips, Nelson, Sewell, Graham, Woodward, Joan, 2010. Technology and organization: essays in honour of Joan Woodward. Emerald, Bingley [u.a.

Licklider, J. C. R., 1960. Man-Computer Symbiosis. IRE Trans. Hum. Factors Electron. HFE-1, 4–11.

Malone, T. W., Laubacher, R., Dellarocas, C., 2010. The Collective Intelligence Genome. MIT Sloan Manag. Rev. Spring 51.

Pree, W., n.d. Meta Patterns - A Mean For Capturing the Essentials of Reusable Object-Oriented Design.

Raes, A., Schellens, T., De Wever, B., Vanderhoven, E., 2012. Scaffolding information problem solving in web-based collaborative inquiry learning. Comput. Educ. 59, 82–94.

Rettberg, S., 2005. All together now: Collective knowledge, collective narratives, and architectures of participation. Digit. Arts Cult.

Sanger, L., n.d. The Early History of Nupedia and Wikipedia: A Memoir - Slashdot [WWW Document].URL http://features.slashdot.org/
story/05/04/18/164213/the-early-history-of-nupedia-and-wikipedia-a-memoir (accessed 9.13.13).

Simon, N., 2010. The participatory museum. Museum 2.0, Santa Cruz, Calif.

The GNU Operating System [WWW Document], n.d. URL http://www.gnu.org/ (accessed 12.30.13).

Weber, S., 2000. The political economy of open source software. BRIE Work. Pap.

Welsh, M. A., Dehler, G. E., 2012. Comb. Critical Reflection and Design Thinking to Develop Integrative Learners. J.Manag.Educ. 37, 771–802.

Wikipedia:Five pillars - Wikipedia, the free encyclopedia [WWW Document], n.d. URL http://en.wikipedia.org/wiki/Wikipedia:Five_pillars (accessed 12.30.13).

Wood, D., Bruner, J. S., Ross, G., 1976. The role of tutoring in problem solving*. J. Child Psychol. Psychiatry 17, 89–100.