Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2010
August 15 – September 18, 2010, Montreal, Quebec, CA

DETC2010/CIE-28272

ANALYSIS OF THE STRUCTURE AND EVOLUTION OF AN OPEN-SOURCE COMMUNITY

*Hao-Yun Huang, Qize Le and Jitesh H. Panchal**
School of Mechanical and Materials Engineering
Washington State University, Pullman, WA 99163 (USA)
*Corresponding Author. E-mail: panchal@wsu.edu

ABSTRACT

Open-source processes are based on the paradigm of self-organized communities as opposed to traditional hierarchical teams. These processes have not only been successful in the software development domain, but are increasingly being used in the development of physical products. In order to successfully adapt open-source processes to product realization there is a need to understand how open-source communities self-organize and how that impacts the development of the products. Towards the direction of fulfilling this need, we present an analysis of an existing open-source community involved in developing a web-based content-management platform, Drupal. The approach is based on the analysis of networks using techniques such as social network analysis, degree distribution, and hierarchical clustering. Openly available information on the Drupal website is utilized to perform the analysis of the community. The data is transformed into two weighted undirected networks: networks of people and networks of Drupal modules. Both the structure of these networks and their evolution during the past six years are studied. The networks are visualized by mapping them into images. Based on the analysis, it is observed that the structure of the Drupal community has the characteristics of a scale-free network, which is similar to many other complex networks in diverse domains. Finally, key trends in the evolution of the networks are identified and the possible explanations for those trends are discussed.

1. FRAME OF REFERENCE: OPEN-SOURCE PROCESSES

During the past two decades, open-source processes have gained significant popularity in the software development domain. Various successful products such as Linux, Apache, and Mozilla have shown that open-source processes can be as successful as the processes followed by traditional organizations. The concept of open-source has not only been used in the software development domain but also been recently implemented in physical product development. Examples include open-source 3D printers [1], electronics prototyping platforms [2], cell phones [3], cars [4, 5], prosthetics [6], machine tools, robots, and other socially-relevant design projects [7]. In physical product development, open-source refers to the openness of the information such as design details, schematics, CAD models, bills of materials, associated software, etc. The success of open-source processes in physical product development is driven by the fact that physical products are also information products during the design phase, and accelerated by the reducing prices of 3D printing capabilities.

Open-source processes emerged in software products earlier than in physical products because software products have characteristics particularly suitable for open-source processes. The development of both software products and physical products can be divided into four phases: design, manufacturing, distribution, and upgrade. In the design phase, both software and physical products can be viewed as information-based products defined of requirements, functions and detail designs, which can be recorded electronically. In the manufacturing phase, software products can be manufactured (programmed) by individuals using computers, while physical products need to be manufactured by specific physical tools and machines. In the distribution phase, software products can be shared through the Internet at (almost) zero cost, but physical products need to be transported from one site to another. In the upgrade phase, the software products can be upgraded by simply re-compiling the upgraded software code whereas physical products need to be re-manufactured. By comparing software and physical products in the four product development phases, it is clear that software products are easier to manufacture, distribute and upgrade than physical products even by individuals without many resources. Hence, open-source processes emerged in software development.

With the increasing availability of 3D printing capabilities, open-source processes have also started gaining popularity in physical product development. In the design phase, physical products are also information-based products whose documents can be easily shared by individuals. So the

Copyright © 2010 by ASME

open-source processes can be applied during the design phase, examples include Open Source Car [4, 5], and Open Prosthetics Project [6]. With the development of the rapid prototyping technology, open source processes can also be extended into product manufacturing. A 3D printer is a convenient tool for individuals to prototype physical products using CAD models. Besides, some physical products such as electronic hardware can be manufactured directly from electronic design documents. These designs can be downloaded from the Internet. In these cases, the manufacturing and upgrading phases can also be carried out through open source processes. As a summary, many physical product development projects have started utilizing open-source processes. In a recent article, Anderson [8] projects that open-source will revolutionize the way in which innovative products are designed and developed. Given the increasing adoption of open-source as a new paradigm for product realization, it is becoming important to understand its underlying dynamics.

Open-source processes are significantly different from traditional product development processes because they are based on bottom-up design by self-organized communities as opposed to top-down design by hierarchical organizations. They are driven by participants choosing their activities based on their own goals and interests instead of being driven by top-down hierarchical control as in the case of traditional product development. Open-source products are always under continuous development and evolution. In traditional product development, the effect of organizational structure on the product is well recognized [9]. According to Conway [10], "any organization that designs a system (defined more broadly than just information systems) will inevitably produce a design whose structure is a copy of the organization's communication structure". Hence, the organizations strive to align the organizational structures with the product structures. However, in the case of open-source processes, no organizational structures are imposed at the beginning of the process. The structure of the organization evolves as new participants join and collaborate with existing participants. The collaboration between different participants is based on the product structure and is driven by the dependencies between subsystems, implying the effect of product structure on community structure. Hence, in open-source processes, the products and communities undergo interdependent co-evolution. In order to successfully utilize open-source processes for product realization, we believe that the knowledge of this interdependent co-evolution is crucial.

The knowledge can be gained by understanding a) the structure and evolution of communities, b) the structure and evolution of products, and c) the interdependence between structures and evolution of communities and products. The focus in this paper is on the first aspect, i.e., understanding the structure and evolution of open-source communities. In the following section, existing literature is discussed and the gap is identified. The proposed approach, involving the analysis of an existing open-source community, is discussed in Section 3. The results from the execution of the approach for a specific open-source community are presented in Section 4. Finally, closing thoughts are presented in Section 5.

2. REVIEW OF EXISTING LITERATURE

Existing literature on open-source processes is primarily focused on open-source software (OSS) development because of highly developed processes, large number of communities and significant amounts of data on OSS development. A general discussion of the factors affecting the success of open-source software development is presented by Weber [11]. OSS is a public good provided by volunteers – the "source code" used to generate the programs is freely available to read, use and modify [12]. An OSS development project is typically initiated by an individual or a small group with ideas which can realize their intellectual, personal, or business interests [13]. Various researchers have presented empirical and quantitative studies on the structure of OSS communities based on the data from existing OSS projects. Raymond [14] describes the Linux development community as the "Bazaar" structure. Cox [15] presents initial thoughts of "town councils" structure in OSS community based on Linux 8086 project. The author conceptually illustrates the community structure for Linux 8086 project. Weber [11] discusses different types of organization structures in various OSS projects. For example, the community structure of the Linux project reflects a pyramid structure whereas the community structure of the BSD project is represented as concentric circles. The structures concluded by Weber are based on direct observation of communities without rigorous mathematical analysis. Crowston and Howison [16] discuss community centralization in OSS development communities by analyzing data from the bug-tracking system in SourceForge. The authors demonstrate that the community centralization or decentralization is not a characteristic of OSS projects. Crowston and Howison [17] later analyze hierarchy and centralization of the OSS communities of Apache, Savannah and SourceForge by employing social network analysis (SNA) metrics. They conclude that large projects are less centralized and hierarchical, as compared to smaller projects. Xu and Madey [18] discuss role distribution and degree distribution in the SourceForge community. Xu and co-authors [19], and Gao and Madey [20] study topological properties of open-source communities, including degree distribution, diameter, clustering coefficient, centrality and component distribution by modeling OSS communities as complex social networks. They also observe small-world [21] and scale-free [22] network properties in the SourceForge community. Xu et al. [23] present the structure of OSS communities by calculating the modularity of the network, which is defined as the fraction of edges within communities minus the expected value of the same quantity if edges fall in a random network, and analyzing the groups that exist in the SourceForge network.

The studies discussed above are focused on analyzing the community structures. Some efforts have also been carried out on the evolution of the communities. White et al. [24] introduced the analysis of social structure over time using snapshots of data. Nakakoji et al. [25] discuss the evolution of communities in the form of role changes of the members in OSS communities, and conclude that there are two factors determining the evolution of OSS communities: the existence of motivated members, and the social mechanisms of

 Copyright © 2010 by ASME

communities. Weiss et al. [26] trace the evolution of a community by taking snapshots of its membership at regular intervals and establish a major hypothesis that OSS communities grow through a process of preferential attachment. de Souza et al. [27] represent a framework for software modules and software developers, and study software project communications at two points in time. The authors analyze the movements of developers across different modules of software systems. Howison et al. [28] investigate the structure of OSS development communities over time using snapshots of data to understand the dynamics of social structures in OSS development communities. They examine three properties of the social structures, namely, centralization, network center, and stability of participation. Wiggins et al. [29] analyze the dynamics of OSS development communities and find a variation in communication centralization and decentralization in the OSS development communities. Open-source software development is a special case of mass-collaborative product development [30]. Panchal [31] presents an agent-based model to model the evolution of products in such bottom-up processes. Panchal later extends the model to explore the co-evolution of communities and products [32]. Le and Panchal [33] study the effect of product architecture on the evolution of products in mass-collaborative processes.

Existing studies are limited in the analysis of open-source communities because of the lack of: a) simultaneous analysis of structure and evolution, b) comprehensive analysis of the different aspects of the community structure, c) trends and patterns in the evolution of communities. Finally, integrated analysis of the evolution of communities and the products has not been carried out. In this paper, we perform a comprehensive study of the community structure and evolution. The study is based on the data from Drupal [34], which is an open-source software for building community-based websites. The reason for studying Drupal is that there is freely available data, detailed documentations, and highly developed community associated with this project. Besides, Drupal is widely used as a basic framework of web development and is a very successful MCPD tool. The PIs are also studying the Sourceforge community to ensure the generality of the results. The objective is to understand the evolutionary characteristics of open source projects that span both software and physical products. The study of open source software project will lead to a) fundamental knowledge which can be applied to both software products and physical products based on their commonalities in the design phase and b) new techniques enabling individuals involved in manufacturing, upgrade and distribution phases.

3. APPROACH ADOPTED FOR THE ANALYSIS OF STRUCTURE AND EVOLUTION OF OPEN-SOURCE COMMUNITIES

The approach adopted is this paper is based on network analysis. The communities are modeled as social networks, defined by participants connected by collaboration links. A social network is defined by a set of interrelated social entities. Social network analysis has been used to analyze diverse systems such as author and paper networks [35], online communities on websites such as Yahoo and Flickr [36], and OSS communities. The overall approach adopted in this paper is highlighted in Figure 1.

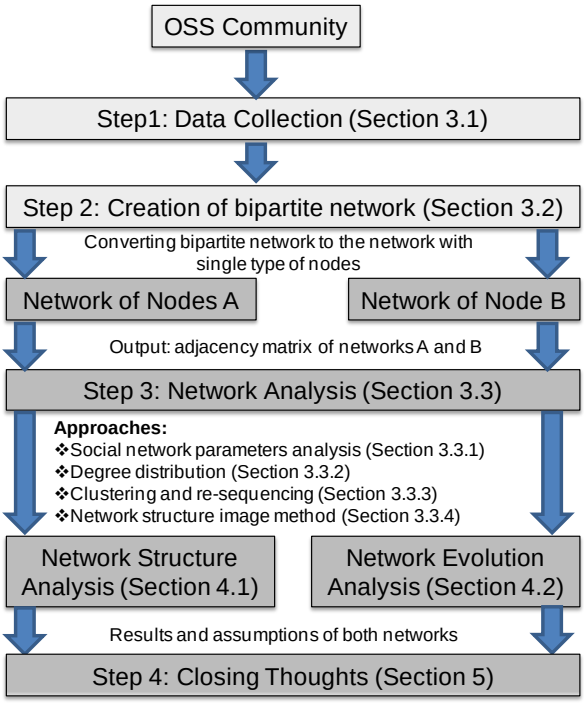

Figure 1 - The approach used to analyze the structure and evolution of open-source communities as networks

3.1. Data Collection

In the first step, raw data about the participants and the product modules they contribute to are extracted from the database. This raw data is used to derive information about the relationships between individuals and their related modules. The raw data can be in the form of a simple table which shows the participants and the modules. In order to study the evolution of the community network, the following information is collected: a) the joining dates of individuals, b) the dates of individuals' first contribution, and c) their contributions to different modules.

Figure 2 (left) is a sample information table from www.drupal.org, which includes information about the participants' activities on a module named "Activity". It contains information about the user name ("User" column), the first and last time each user made a revision of this module ("First commit" and "Last Commit" columns), and the number of times each user revised this project (the "Commits" column). Each Drupal module has an information table similar to Figure 2 (left). After collecting the information from all the modules, the overall information table as shown in Figure 2 (right) is generated.

3.2. Generation of the Networks

From the information table generated in the data collection step, community networks are created to model the relationships within the community. First, we build a network consisting of two types of nodes - people and projects. The development of each module is a project in Drupal. Hence,

Copyright © 2010 by ASME

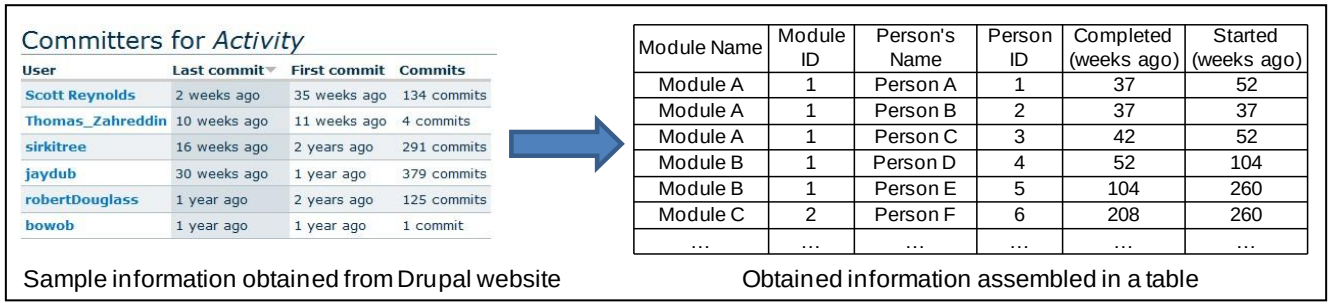

Figure 2 - Illustration of the data-gathering step

each module represents a project node and each participant represents a person node in the network. Such a network is called a bipartite network, $G = \{S_1 \cup S_2, E\}$, which consists of two disjoint sets of nodes S_1 and S_2 and a set of edges E such that each edge in E connects a node in S_1 to a node in S_2. An example of a bipartite graph is shown in Figure 3, where S_1 = {a, b, c, d, e, f, g} and S_2 = {1, 2, 3, 4}. In the case of the open-source software network, assume that S_1 and S_2 represent people and projects respectively.

The links in the network connect a person with a project. Hence, a link represents a person working on a module. The bipartite network can be weighted or binary. If a binary matrix is used, then the links only represent the presence of relationship between people and projects. However, if a weighted network is used, the weights on the links can be used to represent the amount of effort invested by the participants on corresponding modules. An indicator of the amount of effort is the number of commits by a person to a module.

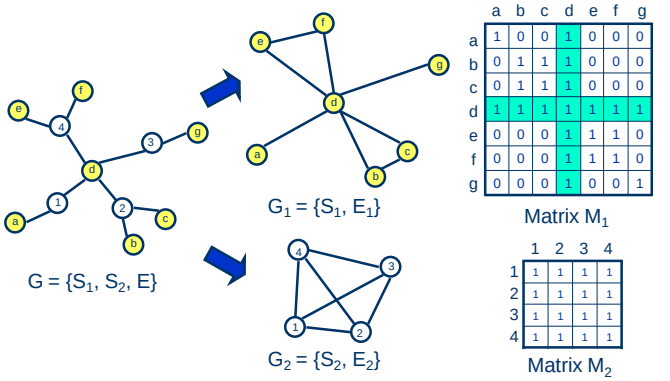

Figure 3 - Example of a bipartite network and the two derived networks

In Figure 3, a binary bipartite graph is illustrated. The bipartite graph G can be transformed into two weighted undirected graphs $G_1 = \{S_1, E_1\}$ and $G_2 = \{S_2, E_2\}$ consisting of people (S_1) and projects (S_2) respectively. Figure 3 provides an illustration of graphs G_1 and G_2 derived from a bipartite graph. Two people in G_1 are connected by an edge if both of them share at least one project. Similarly, two projects in G_2 are linked if they have at least one common participant. The weights associated with edges E_1 represent the number of projects shared by a set of people. Similarly, the weights in graph G_2 represent the number of common participants shared by projects. The graphs can be also represented in matrix form as shown in the figure. An adjacency matrix of a network with n nodes is an n x n matrix, where an element a_{ij} denotes the weight on the edge from node i to node j and 0 denotes no connection between nodes i and j. The diagonal elements are conventionally set to 1. The graphs G_1 and G_2 are used for analysis discussed in Section 3.3.

3.3. Metrics for Network Analysis

The adjacency matrix serves as a basic input for the network analysis process. After creating the adjacency matrix, the network properties are explored using different approaches. In this paper, the following approaches are used to determine the characteristics of the networks: social network metrics, degree distribution, clustering and re-sequencing, and network structure image method. These approaches are discussed in Sections 3.3.1 through 3.3.4.

3.3.1. Social Network Analysis (SNA) Metrics

As discussed earlier, the OSS community is modeled as a social network comprising of participants, projects, and the relationships among participants and projects. In order to characterize the key features of the OSS network, we use Social Network Analysis (SNA) metrics [37]. SNA is a theoretical and methodological paradigm for examining complex social structures [38]. Social networks can be either directed or undirected. The arcs may also carry weights to represent the strengths of the relationships between actors [39]. The following SNA metrics are used in this paper: degree, clustering coefficient, diameter, shortest path, density, connectedness, and degree centrality.

a) Degree is the number of nearest neighbors of a vertex [40]. In an undirected graph, the degree of a vertex v is the number of edges incident with v and is denoted by $deg(v)$ or k_v [41]. The degree distribution, $P(k)$, of a network is defined as the fraction of nodes in the network with degree k [42]. In a bipartite network, two degree distributions corresponding to both types of nodes are important. The joint degree distribution of a network, $P(k_1, k_2)$, represents the probability that a randomly selected edge is connected to nodes with degrees k_1 and k_2 [43, 44]. The joint degree distribution is different from the conditional probability $P(k_2 \mid k_1)$ which measures the probability that a given node of degree k_1 is connected to a node of degree k_2. The metric of degree describes the number of relationships one participant has. The degree distribution represents the fraction of participants in the community with the same number of relationships.

Copyright © 2010 by ASME

b) *Clustering coefficient* is the probability that two nearest neighbors of a vertex are also the nearest neighbors of one another [40]. The clustering coefficient reflects the "cliquishness" of the mean closest neighborhood of a vertex. Large clustering coefficient indicates the rate at which information can be spread in the community.

c) *Diameter* is the largest distance between any two nodes of a connected network [41]. The diameter of a network indicates how "big" the network is. Hence, the metric of diameter provides information about how large the community is.

d) *Shortest path* is the shortest path of vertices and edges that links two vertices in a network [45]. The average shortest path can describe whether there exists the "small world phenomenon" [46] within the community.

e) *Density* of a network is the average proportion of links incident with nodes in the network [47]. The density of a network ranges from 0 (if there are no links present) to 1 (if all possible links are present). A network with density of 1 is also called a complete network. The density of a network with *n* nodes and *m* links is:

$$density = \frac{2m}{n(n-1)}$$

High density of the community means that on average each participant has a large number of relationships that enable participants to communicate with each other.

f) *Connectedness* represents the ratio of the number of pairs in the directed graph that are reachable relative to the number of ordered pairs.

g) *Degree Centrality* measures the degree of inequality or variance in the network as a percentage of that of a perfect star network of the same size. For a network G=(V,E) with n nodes, the degree centrality $C_D(v)$ for a node v is [48],:

$$C_D(V) = \frac{\text{degree (v)}}{n-1}$$

The degree centrality of a network G is:

$$C_D(G) = \frac{\sum_{i=1}^{n}[C_D(v*) - C_D(v_i)]}{n-2}$$

where $C_D(v*)$ is the highest degree centrality of a node in the network. $C_D(v_i)$ is the degree centrality of a node i in the network. The degree centrality of the entire community is also called degree centralization, which measures the inequality of the relationships among participants in the community.

Social network analysis tools are used for evaluating the metrics discussed above. The most widely used tools are Structure [49], Gradap [50], UCInet [51] and Network Workbench [52]. Other social network analysis tools are discussed by Huisman et al. [53] and Freeman [48]. UCInet is used for the results presented in this paper.

3.3.2. Degree Distribution and Scale-Free Network

The degree of a node is the number of links connected to it. The degree distribution is the possibility distribution of all the degrees in the entire network. Recent research has found that degree distribution in many real-world networks satisfies a power law [54], $y=bx^a$ where *b* and *a* are constants and *y* denotes the number of nodes with degree *x*. A network whose degree distribution follows a power law is called scale-free network. Scale-free networks have a property that only a few of nodes (called "hubs") have a high degree, while most other nodes are only connected to a few nodes. Scale-free networks have different characteristics as compared to random networks.

3.3.3. Hierarchical Clustering

Clustering is an approach for assigning a set of objects into subsets (clusters) such that the objects within a cluster are closer to each other as compared to the objects in different clusters [55]. Hierarchical clustering involves recursive clustering using previously assigned clusters [56]. At the highest level of the hierarchy, all objects are within one cluster. At the lowest level, each object is its own cluster and the number of clusters is equal to the number of objects. Between the highest and lowest levels, various intermediate levels of clusters are generated based on the similarity (or closeness) or distance between different objects. Various measures such as Euclidean distance, Manhattan distance, maximum distance, Mahalanobis distance, and cosine similarity are commonly be used. Hierarchical clustering is used in statistical data analysis, pattern recognition, and data mining applications. In weighted networks, the weights can be used to represent the similarity or dissimilarity between nodes. For the OSS social networks discussed in this paper, the weights represent the closeness between people and modules. Clusters of people represent participants working closely with each other. The result of hierarchical clustering is a tree with closely related nodes closer to each other and the dissimilar nodes distant from each other. The relative sizes of clusters and their overlap convey significant information about the network structure.

3.3.4. Network Structure Image Method

For complex networks with a large number of nodes, the visualization of the clusters as nodes and links is difficult. An alternate visualization approach is based on the image representation of a matrix. The image corresponding to an adjacency matrix provides a convenient visual representation of the network structure. A network structure image is generated by mapping each element of the matrix into a point in the image. Hence, an n x n matrix maps into an image with a size of n x n pixels. The color of a point in the image corresponds to the values of the elements in the matrix. This image method is usually combined with the hierarchical clustering method to visualize the clusters within networks.

4. CASE STUDY - DRUPAL NETWORK

The approaches discussed in the previous section are utilized to analyze the structure and evolution of Drupal

Copyright © 2010 by ASME

community (www.drupal.org). Drupal is an open-source content-management system, which allows the creation of community-based websites. The Drupal framework consists of a core and a large number of modules developed by users using open-source techniques. Drupal has been chosen for this study because of its strong community and the easy access to participant and module data for analysis. The Drupal project was started in 2000 and it currently has a large community of contributors. There are different types of users who interact with the Drupal community. *Passive Users* are users who only download and use the software but do not contribute to the code. *Active Users* contribute to the discussion board and identify bugs but do not modify the code. *Co-developers* modify the codes, fix bugs, and add new features to the software. *Core developers* contribute largely to the core of the Drupal code and coordinate co-developers' work. *Project leaders* are the project administrators who manage the direction of the entire project. For the analysis presented in this paper, only co-developers, core developers, and project leaders are considered. Activities such as the identification of bugs and contributions to discussions on the bulletin boards are not considered in this paper. The bipartite network created for the analysis of Drupal community consists of two types of nodes – a) the people (participants), and b) the projects (modules) they contribute to. The analyses discussed in Section 3 are carried out for the Drupal data. The analysis of the structure of networks is discussed in Section 4.1 and the analysis of network evolution is presented in Section 4.2.

4.1. Analysis of Network Structure

4.1.1. General Discussion of the Network Data

The data was collected for Drupal 5.x in August 2009. The data consists of 1907 projects (modules) and 1217 participants who contributed the code during the nine years from the start of the project in the year 2000. The data is used to create the bipartite network consisting of people and project from which two networks, discussed in Section 3.2, are created. The two networks are referred to as *people network* and *project network*. The characteristics of the two networks are listed in Table 1.

Table 1 - Characteristics of the People and Project networks

Network	Average Degree	Central-ization	Average Distance	Average Density	Clustering Coefficient	Connect-edness
People	24.62	1.26%	2.86	0.0202	0.74	0.4465
Project	42.14	2.71%	2.87	0.0221	0.83	0.5454

It is observed that both the networks are similar in terms of the metrics listed in the table. For networks with over 1000 nodes, the average distance between the nodes of 2.86 and 2.87 are very low. The degree centrality of both networks is also low. The average degrees of the nodes in the two networks are of 24.62 and 42.14. With the low average degrees, we can assume that both networks are in a low-scale unitary connection. The low average density in both networks implies that different people develop most of the projects, and there are more co-developers than core developers. From the table, it is also observed that both the networks have low average distances and high clustering coefficients. The combination of low average distance and high clustering coefficient denotes that this network is highly connected. In network analysis, this is called "small-world" phenomenon. Small world phenomenon means that any two individuals in the network are likely to be connected through a short sequence of intermediate acquaintances [46]. With over a million possible links in people network and three millions in project network, the connectedness, 0.4465 in people network and 0.5454 in project network, show high extent of connectivity of the two networks. These characteristics provide basic information about the network structure. Further details are obtained by degree distribution and clustering in the following sections.

4.1.2. Degree Distribution of the Networks

The degree distributions are plotted in Figure 4 and Figure 5. Figure 4 contains the degree distributions of the nodes in the bipartite network whereas Figure 5 contains the degree distributions of the people and project networks. The points in Figure 4 represent the number of people working on different modules and the number of modules on which different people contribute to. On the other hand, the points in Figure 5 represent the number of projects linked to other projects through common contributors (left) and the number of people linked to other people (right). The X and Y axes in both the figures are the degree cardinality and the number of nodes in different degree cardinality respectively. It is observed that the degree distributions of the networks are linear on a log-log scale indicating a scale-free topology of all three networks. Such a scale-free topology has been observed in many biological, technical, and social networks. The community of Sourceforge has also been shown to have a similar degree distribution.

In a scale free network topology, there is a small set of nodes with a large number of links with other nodes, and a large number of nodes with small number of connections. The nodes with a large number of connections are called the "hubs". The hubs in the project network are the key projects that provide the core functions of Drupal. The hubs in people network are the small number of core developers who communicate with and support a large number of other participants. In the bipartite network, the "hubs" can be either core developers or key projects.

A widely accepted model for generating a scale-free network is the preferential attachment model. According to this model, networks grow through the addition of nodes. New nodes preferentially attach to other nodes with high degree. Hence, the probability of attachment of a new node to existing nodes is proportional to the degrees of the existing nodes. We believe that the model explains the emergence of scale-free networks in the open-source domain because the modules that have higher number of participants develop faster, thereby increasing the modules' utility, and hence attract even more participants.

Copyright © 2010 by ASME

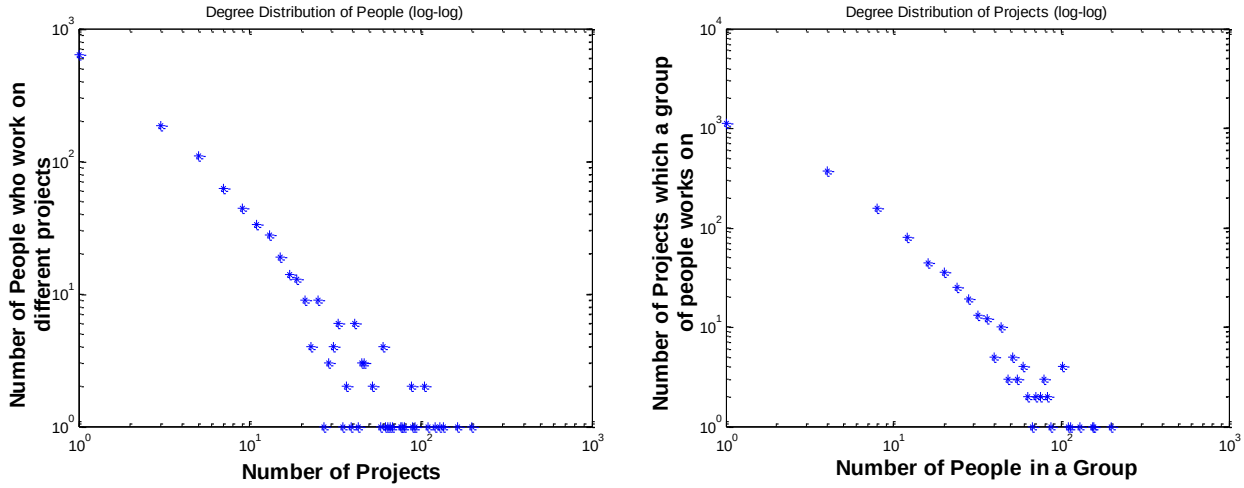

Figure 4 - Degree distribution of two types of nodes in the bipartite network

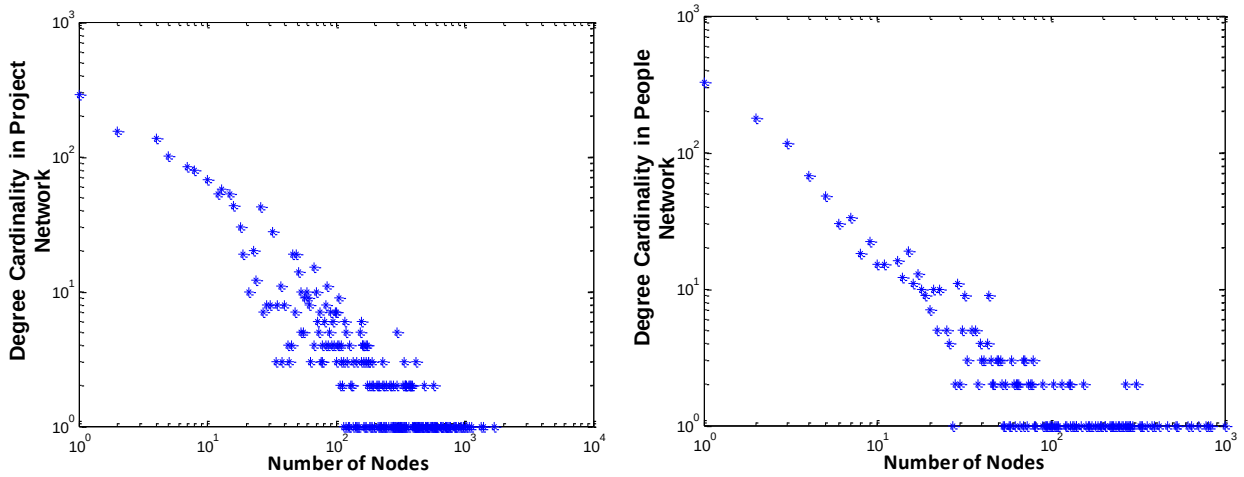

Figure 5 - Degree distribution of projects (left) and people (right) networks

4.1.3. Network Structure Analysis through Clustering

In this section, we analyze the structure of the network using clustering techniques. The original and clustered networks are visualized using images corresponding to the adjacency matrices of the networks. Figure 6 is a black and white image corresponding to the binary bipartite network with people and project nodes. The horizontal and vertical axes correspond to people and projects respectively. Each pixel in the image represents a pair of nodes. The weight of a link between the pair of nodes corresponds to a color in the image. For example, in Figure 6, black color represents no link between the nodes and white color represents a link with weight = 1, indicating a link between a person and a project. The people and projects are arranged in the increasing order of their IDs.

3D plots are used to visualize the adjacency matrices corresponding to the weighted people and project networks. The plots are shown in Figure 7(a) and Figure 8 (a) respectively. The z-axis corresponds to the weights of links connecting nodes on x and y axes. The weights on the links are used as the similarity measure for clustering purposes. This is because the larger weights on people networks indicate that people are working together on greater number of projects. Similarly, larger weights on the project networks indicate that the projects share greater number of participants. Hence, the larger the weights, closer the nodes are.

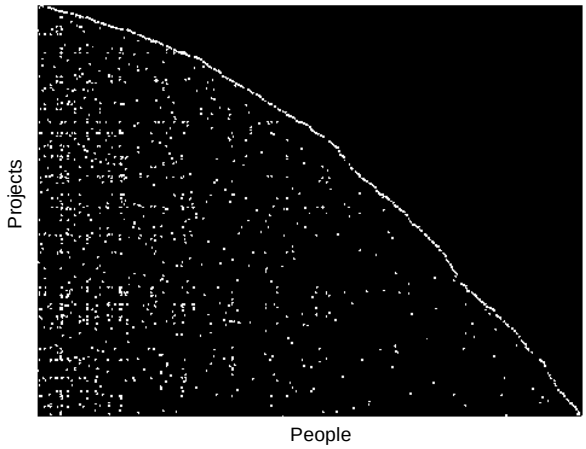

Figure 6 - Adjacency Matrix Image of the Bipartite Network

The adjacency matrices are clustered and the corresponding plots are shown in Figure 7(b) and Figure 8(b).

Copyright © 2010 by ASME

From these plots of the clustered networks, it is clear that there are few participants near the bottom-right corner who have connections with a lot of other people. These participants are the core developers who contribute to the entire project and oversee the work of other participants. These participants are also highly connected with each other. The other participants are weakly connected and have a few links with each other. As we discussed in the previous section, this kind of distribution is due to the scale free nature of the network. Hence, the plots provide an indication of the different roles of the individuals. Similar characteristic is also observed in the project network shown in Figure 8.

In addition to the highly skewed distribution of links between nodes, we also observe clusters of nodes appearing as blocks in the images. The blocks indicate sets of participants or projects that are highly connected with each other. The highly connected participants are similar to teams of individuals working together in traditional product realization. The difference, however, is that these clusters (teams) in open-source are emergent as opposed to being pre-defined as in traditional product realization.

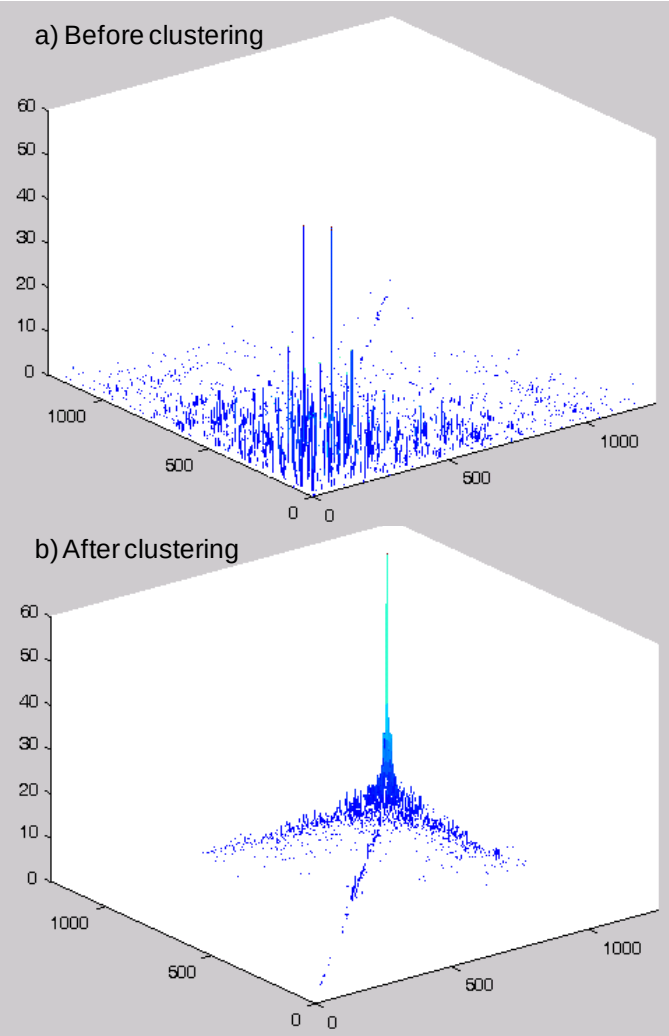

Figure 7 - Adjacency matrix plots of people network before and after clustering

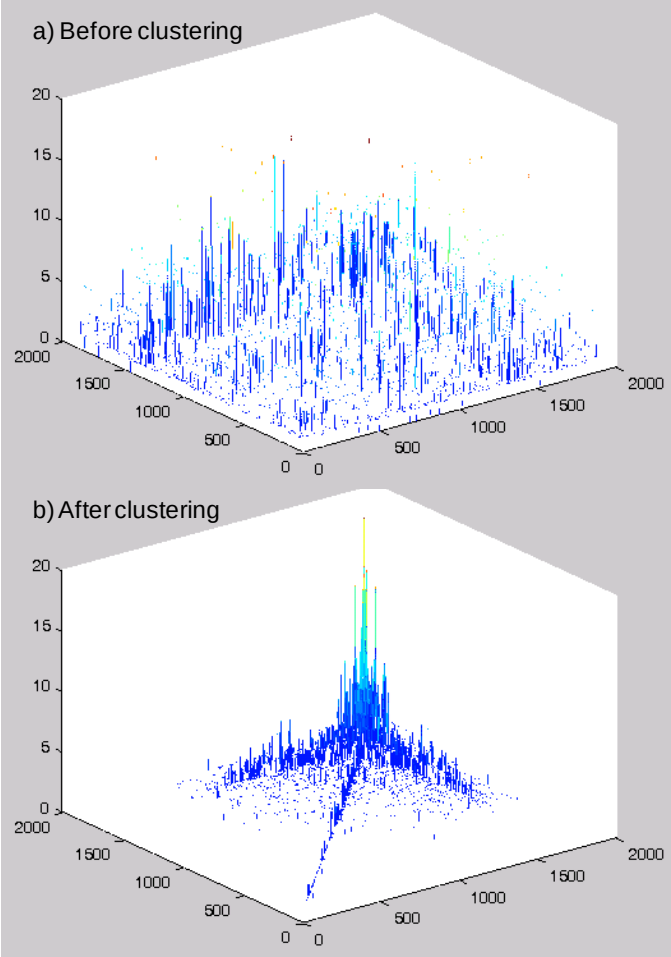

Figure 8 - Adjacency matrix image of project network before and after clustering

4.2. Analysis of Network Evolution

In Section 4.1, we discussed network structure of Drupal using the data from August 2009. Drupal was founded in 2000, rapidly growing from 12 developers and 23 projects to 1217 developers and 1907 projects. The network evolution analysis is aimed at understanding how the bipartite, people, and project networks grow over time. From the analysis of network evolution, our objective is to identify characteristics of the network that change over time and the characteristics that are invariant with time. The knowledge about the evolution of these networks can be used to direct community growth.

4.2.1. Generation of Snapshots of Networks at Different Intervals

During nine years of development, Drupal community has grown by a factor of about 100, both in terms of the number of participants and modules. In order to analyze the evolution of the community, six snapshots of the data are generated based on the time when people joined the community and time when projects were created. The snapshots are generated at intervals of one year, starting with year 4. Different snapshots of data are not generated for the first three years because most of the evolution in the network took place between years 4 and 9 of the project. The number of participants (people) and modules

Copyright © 2010 by ASME

(projects) at different snapshots are shown in Figure 9. It is observed that the number of participants and modules has grown exponentially. The exponential curve fit for both is shown in the figure, which provides an important indicator of evolution of the Drupal community.

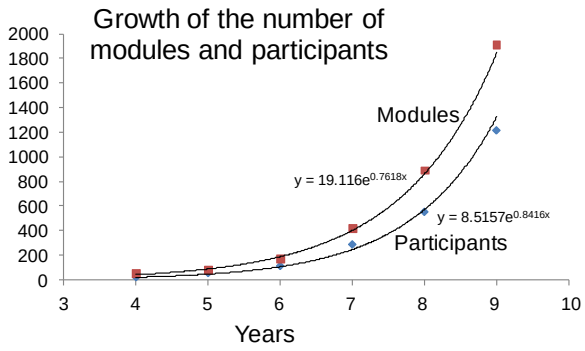

Figure 9 - Number of people and modules at different times

4.2.2. Evolution of the Network Characteristics

The social network metrics are used to analyze the characteristics of the six snapshots of the people and project networks. The results are shown in Table 2. The trends in the evolution of the networks are analyzed using average degree, degree centrality, average density, clustering coefficient and connectedness.

In Figure 10(a), the average degree of the nodes in the networks is plotted at different time steps. The plots show three stages in the networks' evolution - the first stage where the average degree increases, the second stage during which the average degree reduces, and finally the third stage where the mean degree remains relatively constant. The three stages are observed in both product network and the project network. We believe that this trend can be explained as follows. During the first phase, only the core developers contribute to the project. All the developers closely collaborate with each other. Any addition to the number of core developers results in the increase in the degree of all the nodes in the network. Hence, the average degree increases during this first phase. This phenomenon has also been discussed as an initial euphoria among the co-developers and core-developers [36]. During the second phase, as more developers join the network, the number of participants and modules (i.e., nodes in the network) increase at a rate faster than the number of links. Hence, the mean degree reduces. During the third phase, the existing participants explore other modules and initiate collaborations with other participants.

Figure 10(b) shows the degree centrality of the network as a function of time. The degree centrality of both the networks shows a steep decline during the initial phase, indicating the significant difference between the nodes with highest degree and the other nodes. During the initial years of an open-source project, core-developers play an important role in the development of the project. At that time, some project leaders start lots of projects while others start only a few. This causes the significant difference seen in the figure. During the later phases, there are more choices of projects for people to contribute to, which results in a gradually decreasing curve.

Table 2 - Highlights of the Drupal network evolution

Network	Year	Average Degree	Degree Centrality	Average Density	Clustering Coefficient	Connect-edness
Projects	9	54.5	0.2034	0.7174	0.868	0.8254
	8	91.7	0.1513	0.5559	0.912	0.7116
	7	57.9	0.0810	0.1392	0.853	0.6148
	6	45.8	0.0429	0.0517	0.838	0.5981
	5	39.2	0.0307	0.0262	0.841	0.4945
	4	42.1	0.0271	0.0221	0.827	0.5454
People	9	24.0	0.1405	0.4800	0.842	0.7420
	8	32.7	0.0561	0.3031	0.855	0.6959
	7	32.8	0.0341	0.1155	0.816	0.5685
	6	30.3	0.0227	0.0553	0.787	0.5084
	5	24.8	0.0157	0.0277	0.754	0.3943
	4	24.6	0.0126	0.0202	0.740	0.4465

The average density of the networks is plotted in Figure 10(c). The trend of the average density is similar to the degree centrality. It reduces steeply during the initial phase and then reduces gradually during the later phases. During the initial phase, there is a closely connected group of core-developers and project leaders, which means that the nodes are highly connected in both project and people networks. In order to maintain the same density of a network, the number of links must increase at a rate equal to the square of the number of nodes. However, we observe that rate of increase of the number of links is almost equal to the rate of increase in nodes. Hence, the density reduces as shown in Figure 10(c).

The clustering coefficients of the people and project networks are plotted in Figure 10(d). After reaching their peaks in a short time, both the curves undergo a steady decline. We believe that the reason for the downtrend is the exponential increase in participants in the Drupal network. Further, the core modules of Drupal were developed during the first phase, resulting in the high clustering of the people network. The closely related core modules result in higher initial clustering coefficients. Despite its decline, the clustering coefficients in both networks are still high compared to random networks. The low average node distance in both networks combined with the high clustering coefficient is a characteristic of scale-free and small-world networks.

4.2.3. Evolution of Scale-free Property

In Section 4.1.2, we discuss the degree distribution of the two networks. In this section, we explore how the degree distributions of the Drupal network change over time. The degree distributions of the participant network for three snapshots of data are shown in Figure 11.

It is observed that during each data snapshot, the form of the degree distribution remains the same – power law. The only difference is in terms of the parameters of the power law. The coefficient increases while the exponent decreases with time as shown in the figure. Faloutsos [57] has shown that the frequency, f of an outdegree, d, is proportional to the out-degree to the power of a constant k.

Copyright © 2010 by ASME

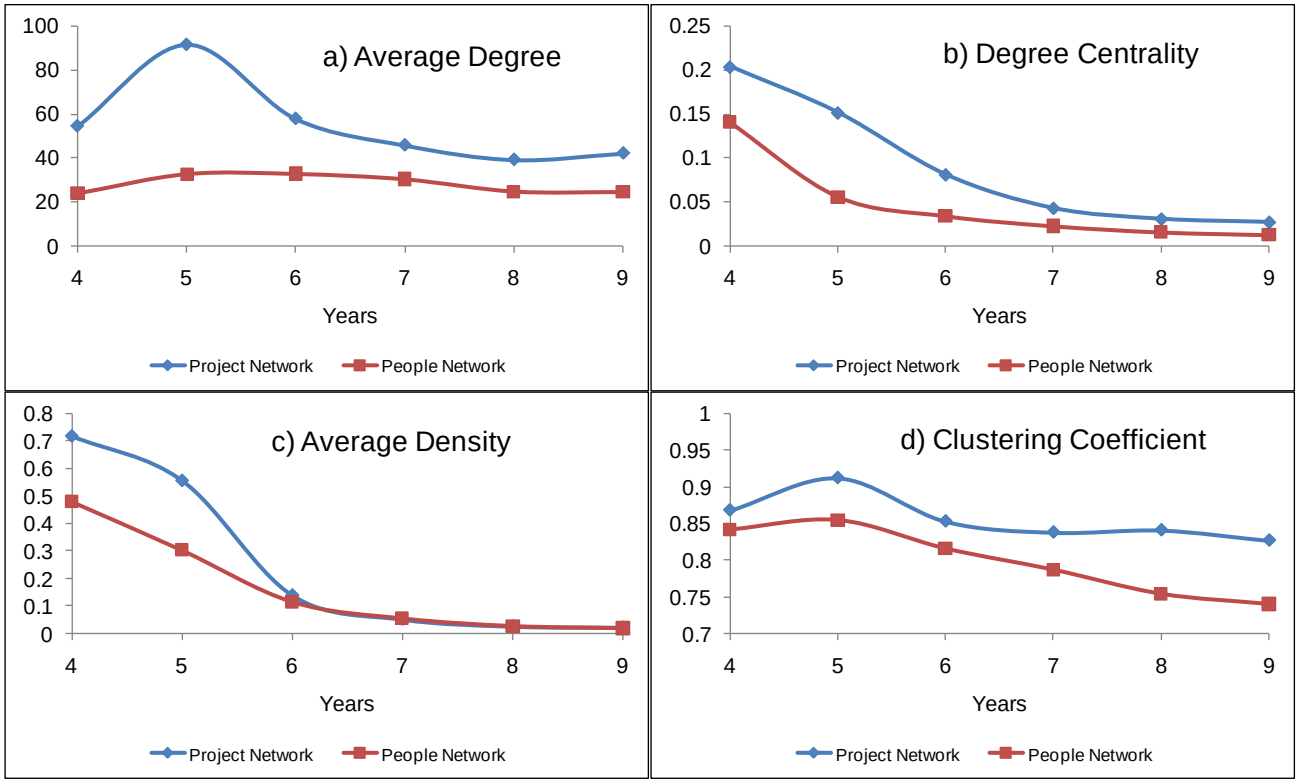

Figure 10 - Average degree, degree centrality, average density, and clustering coefficients of Project and People Networks with respect to time

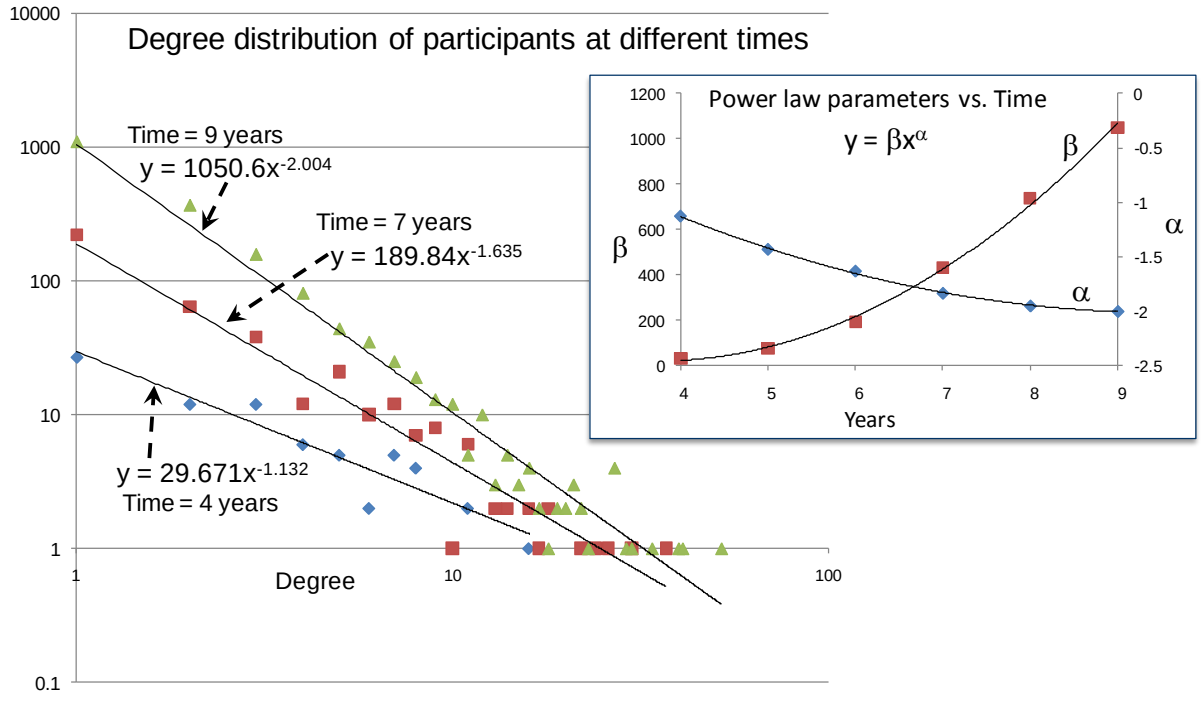

Figure 11 - Evolution of the power-law coefficients for the people network

Copyright © 2010 by ASME

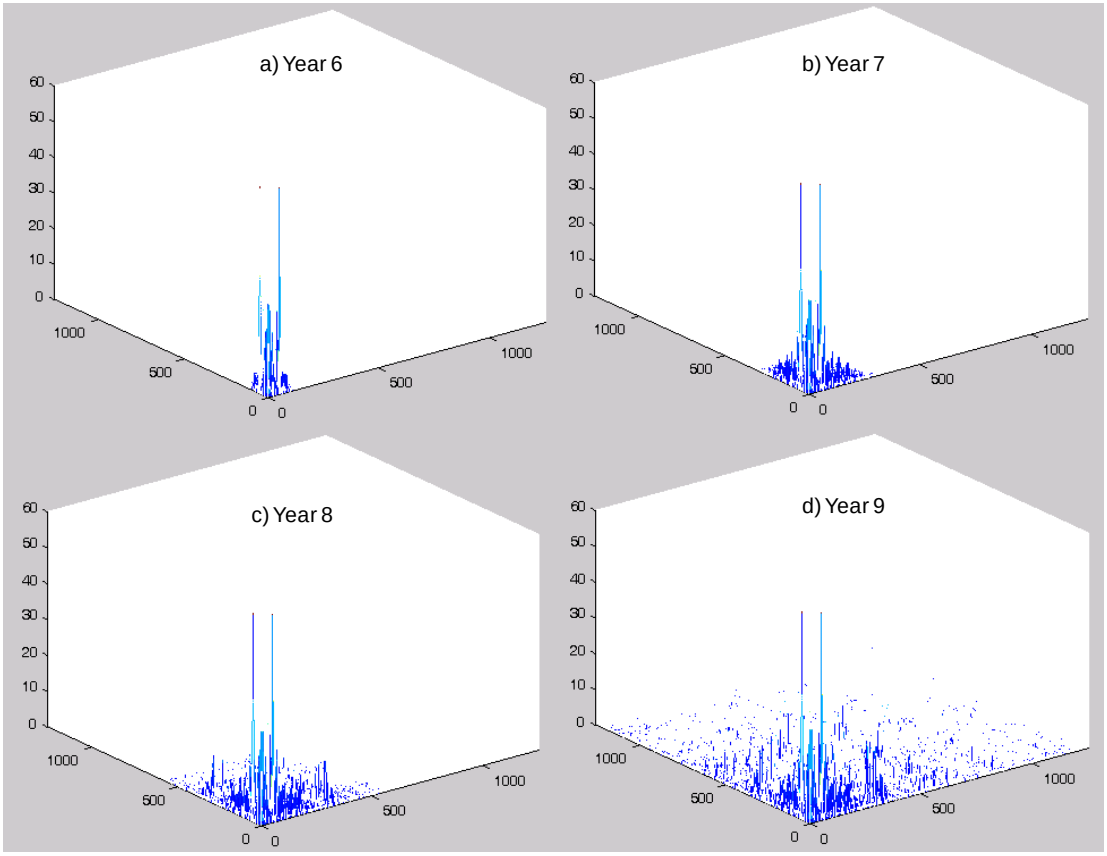

Figure 12 - Adjacency Matrix Image of Evolution of People Network

From Figure 11, we have obtained the relationship from our data, the frequency and degree:

$$f = \beta \times d^{\alpha}$$

where d is the degree and f is the frequency of a degree, α and β are constants.

It is observed that there is an approximate exponential relationship between α and β in bipartite network. This helps in eliminating one constant α or β in the equation of frequency and degree. Hence, the evolution of the Drupal network far from arbitrary and follows some common trends observed in complex networks [57]. Currently, we have obtained the equation of frequency and degree with only one constant. In future work, we will determine whether there is a relationship between the constant and other parameters of the network, such as the number of nodes or links of the network.

4.2.4. Analysis of Network Structure Evolution

In this section, we present the adjacency matrix for the snapshots of network at different times. In both people and project networks the row and column sequences, which stand for participants and projects are ordered for Years 6-9. This method is effective for observing the evolution of the network structure. Figure 12 shows the adjacency matrices of the people network at different times as 3D plots. The participant IDs are organized in an increasing order based on the time when they joined Drupal project. Hence, the developer who joined Drupal four years ago has an ID greater than the participant who joined two years ago. Similarly, the projects' IDs are also arranged in increasing order.

The heights of the bars in Figure 12 indicate the values in the adjacency matrix. In Figure 12(a), we observe that almost every participant has strong collaborations with others. In the second plot, more participants are added and the strengths of collaboration between newly added participants are weaker. The strengths of collaboration between existing participants continue to become stronger, as indicated by the change in heights of bars corresponding to existing participants. Finally, the plot corresponding to Year 9 shows the presence of small groups of participants who collaborate with almost everyone in the network, which signifies that these are participants who joined later but became core developers. Similar trends are observed in the projects network displayed in Figure 13. The block close to the origin refers to the core modules. The height corresponding to the core modules increases as more and more modules are added.

5. CLOSING THOUGHTS

Open-source software development community is different from a closed-source software development team. Participants in open-source software development community collaborate equally based on their own interests, while close-source software development team is formed in hierarchical structures in which tasks are clearly defined and people are needed to report to their direct supervisor. In this paper, an analysis of the structure and evolution of an open-source software development community is presented. The primary objective of this analysis is to understand patterns in community structures and trends in their evolution. An

Copyright © 2010 by ASME

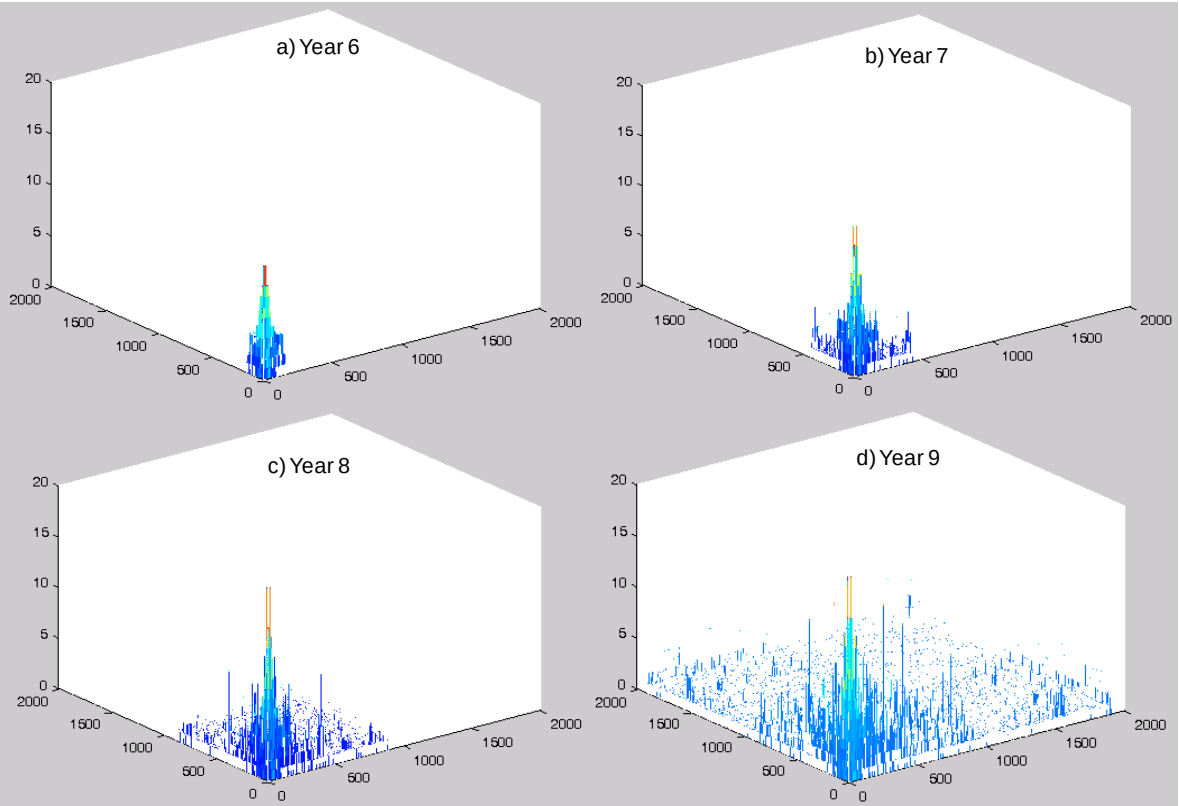

Figure 13 - Adjacency Matrix Image of Evolution of Project Network

understanding of such patterns and trends, and their impact on the product evolution can potentially be used for guiding open-source projects. We believe that the patterns of community structures are similar in both open-source software and hardware. The analysis is carried out using existing open-source software development communities because of the availability of data from successful projects. Based on the analysis, it was observed that community structures display scale-free characteristics, which have been observed in a large number of other social, technical and biological networks. The hypothesis is that the scale-free network topology is a result of preferential attachment of individuals to well-developed projects. In this paper, the changes in the scale-free characteristics of the network are also discussed.

Further research in this area would involve developing a simulation model to validate the hypothesis by developing the community network in a bottom-up manner using local decisions. The model to simulate the growth of community will based on attachment kernel, which is defined as the probability that a newly introduced participant links to an existing participant which already has k links. Parameters will include initial number of people and projects, probability of starting a new project p, and k_0 which is a constant based on Krapivsky and Redner's model [58].

6. REFERENCES

1. Olliver, V., 2005, *Construction of Rapid Prototyping Testbeds Using Meccano,* [cited January 21, 2010]; Web Link: http://staff.bath.ac.uk/ensab/replicator/Downloads/MeccanoFDMfinal.pdf.
2. Oxer, J. and Blemings, H., 2009, *Practical Arduino: Cool Projects for Open Source Hardware*, New York, NY: APress.
3. OpenMoko, 2008, *OpenMoko,* [cited April 11, 2008]; Web Link: http://www.openmoko.com/.
4. Oscar Project, 2008, *Oscar: Reinvent Mobility,* [cited 8 February 2008]; Web Link: http://www.theoscarproject.org/.
5. Rally Fighter, *Product Brochure,* [cited February 05, 2010]; Web Link: http://www.local-motors.com/assets/rally-fighter-brochure.pdf.
6. Open Prosthetics, 2008, *The Open Prosthetics Project: An Initiative of the Shared Design Alliance,* [cited April 11, 2008]; Web Link: http://openprosthetics.org/.
7. Sawhney, N., Griffith, S., Maguire, Y., and Prestero, T., 2002, "ThinkCycle: Sharing Distributed Design Knowledge for Open Collaborative Design," *International Journal of Technologies for the Advancement of Knowledge and Learning (TechKnowLogia)*, **4**(1), pp. 49-53.
8. Anderson, C., February 2010, *In the Next Industrial Revolution, Atoms are the New Bits*, *Wired Magazine*.
9. Sosa, M. E., Eppinger, S. D., and Rowles, C. M., 2004, "The Misalignment of Product Architecture and Organizational Structure in Complex Product Development," *Management Science*, **50**(12), pp. 1674-1689.

Copyright © 2010 by ASME

10. Conway, M. E., 1968, "How do Committees Invent," *Datamation*, **14**(5), pp. 28-31.
11. Weber, S., 2004, *The Success of Open Source*: Harvard University Press.
12. Bessen, J. E., 2006, "Open Source Software: Free Provision of Complex Public Goods", in *The Economics of Open Source Software Development*, (J. Bitzer and P.J.H. Schroder, Editors), Elsevier: Amsterdam.
13. von Hippel, E. and Krogh, G. v., 2003, "Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science," *Organization Science*, **14**(2), pp. 209-223.
14. Raymond, E. S., 2001, *The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*: O'Reilly Media, Inc.
15. Cox, A., 1998, *Cathedrals, Bazaars and the Town Council,* [cited January 11, 2010]; Web Link: http://slashdot.org/features/98/10/13/1423253.shtml.
16. Crowston, K. and Howison, J., 2005, "The Social Structure of Free and Open Source Software Development," *First Monday (online only)*, **10**(2).
17. Crowston, K. and Howison, J., 2006, "Hierarchy and Centralization in Free and Open Source Software Team Communications," *Knowledge, Technology, and Policy*, **18**(4), pp. 65-85.
18. Xu, J. and Madey, G., 2004, *Exploration of the Open Source Software Community*, *NAACSOS 2004*: Pittsburgh, PA.
19. Xu, J., Christley, S., and Madey, G., 2006, "Application of Social Network Analysis to the Study of Open Source Software", in *The Economics of Open Source Software Development*, (J. Bitzer and P.J.H. Schröder, Editors), Elsevier: Amsterdam.
20. Gao, Y. and Madey, G., 2007, "Network Analysis of the SourceForge.net Community", in *The Third International Conference on Open Source Systems (OSS 2007), IFIP WG 2.13*, Limerick, Ireland.
21. Watts, D. J. and Strogatz, S., 1998, "Collective Dynamics of 'Small-World' Networks," *Nature*, **393**, pp. 440-442.
22. Barabasi, A.-L. and Albert, R., 1999, "Emergence of Scaling in Random Networks," *Science*, **286**(5439), pp. 509-512.
23. Xu, J., Christley, S., and Madey, G., 2005, "The Open Source Software Community Structure", in *NAACSOS2005*, Notre Dame, IN.
24. White, H. C., Boorman, S. A., and Brieger, R. L., 1976, "Social Structure from Multiple Networks I. Blockmodels of Roles and Positions," *American Journal of Sociology*, **81**(4), pp. 730-780.
25. Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., and Ye, Y., 2002, "Evolution Patterns of Open-Source Software Systems and Communities", in *IWPSE '02: Proceedings of the International Workshop on Principles of Software Evolution*, New York, NY, ACM Press, pp. 76-85.
26. Weiss, M., Moroiu, G., and Zhao, P., 2006, "Evolution of Open Source Communities", in *Proceedings of the International Conference on Open Source Systems*, Springer, pp. 21-32.
27. de Souza, C., Froehlich, J., and P., D., 2005, "Seeking the Source: Software Source Code as a Social and Technical Artifact", in *GROUP '05 ACM*, Sanibel Island, pp. 197-202.
28. Howison, J., Inoue, K., and Crowston, K., 2006, "Social Dynamics of Free and Open Source Team Communications", in *Proceedings of the IFIP 2nd International Conference on Open Source Software*, Boston, USA.
29. Wiggins, A., Howison, J., and Crowston, K., 2008, "Social Dynamics of FLOSS Team Communication across Channels", in *Proceedings of the Fourth International Conference on Open Source Software (IFIP 2.13)*, Milan, Italy.
30. Panchal, J. H. and Fathianathan, M., 2008, "Product Realization in the Age of Mass Collaboration", in *ASME Design Automation Conference*, New York City, NY, USA. Paper Number: DETC2008-49865.
31. Panchal, J. H., 2009, "Agent-based Modeling of Mass Collaborative Product Development Processes," *Journal of Computing and Information Science in Engineering*, **9**(3), pp. (031007)1-12.
32. Panchal, J. H., 2009, "Co-Evolution of Products and Communities in Mass-Collaborative Product Development - A Computational Exploration", in *International Conference on Engineering Design (ICED'09)*, Stanford, CA. Paper Number: ICED'09/147.
33. Le, Q. and Panchal, J. H., 2009, "Modeling the Effect of Product Architecture on Mass Collaborative Processes - An Agent-based Approach", in *2009 ASME International Design Engineering and Technical Conferences, Computers and Information in Engineering*, San Diego, CA. Paper Number: DETC2009/CIE-86798 (under review for the *Journal of Computing and Information Sciences in Engineering*).
34. Drupal, 2008, *Drupal: Community Plumbing,* [cited July 20, 2008]; Web Link: http://drupal.org/.
35. Börner, K., Maru, J. T., and Goldstone, R. L., 2004, "The Simultaneous Evolution of Author and Paper Networks," *Proceedings of National Academy of Sciences (PNAS)*, **101**(Supplement 1), pp. 5266-5273
36. Kumar, R., Novak, J., and Tomkins, A., 2006, "Structure and Evolution of Online Social Networks", in *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia, PA.
37. Newman, M. E. and Girvan, M., 2004, "Finding and Evaluating Community Structure in Networks," *Physics Review, E*, **69**(2), pp. 026113.
38. Emirbayer, M. and Goodwin, J., 1994, "Network Analysis, Culture, and the Problem of Agency," *The American Journal of Sociology*, **99**(6), pp. 1411-1454.

Copyright © 2010 by ASME

39. Hanneman, R. A. and Riddle, M., 2005, *Introduction to Social Network Methods,* [cited 2008 January 12]; Web Link: http://faculty.ucr.edu/~hanneman/.
40. Dorogovtsev, S. N. and Mendes, J. F. F., 2002, "Evolution of Networks," *Advances in Physics*, **51**(4), pp. 1079-1187.
41. Chartrand, G. and Zhang, P., 2005, *Introduction to Graph Theory*, New York: McGraw Hill.
42. Newman, M. E. J., 2003, "The Structure and Function of Complex Networks," *SIAM Review*, **45**(2), pp. 167–256.
43. Mahadevan, P., Krioukov, D., Fomenkov, M., Dimitropoulos, X., Claffy, K. C., and Vahdat, A., 2006, "The Internet AS-Level Topology: Three Data Sources and One Definitive Metric," *ACM SIGCOMM Computer Communication Review*, **36**(1), pp. 17-26.
44. Zhou, S. and Mondragón, R. J., 2007, "Structural Constraints in Complex Networks," *New Journal of Physics*, **9**(6), pp. 173(1-11).
45. Newman, M. E. J., 2001, "Scientific Collaboration Networks. II. Shortest Paths, Weighted Networks, and Centrality " *Physical Review E*, **64**(1), pp. 016132.
46. Kleinberg, J., 2000, "The Small-World Phenomenon: An Algorithm Perspective", in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing* Portland, Oregon, United States, pp. 163 - 170.
47. Wasserman, S., and Katherine Faust, 1994, *Social Network Analysis: Methods and Applications*, Cambridge: Cambridge University Press.
48. Freeman, L. C., 1988, "Computer Programs and Social Network Analysis," *Connections*, **11**, pp. 26-31.
49. Bur, R. S., 1982, *Toward a Structural Theory of Action: Network Models of Social Structure, Perception and Action*, New York: Academic Press.
50. Sprenger, K. J. A. and Stokman, F. N., 1995, *GRADAP: Graph Definition and Analysis Package, Groningen, ProGamma,* [cited.
51. Borgatti, S. P., Everett, M.G. and Freeman, L.C., 2002, *Ucinet for Windows: Software for Social Network Analysis*, Analytic Technologies: Harvard, MA.
52. NWB Team, 2006, *Network Workbench Tool, Indiana University, Northeastern University, and University of Michigan,* [cited June 10, 2009]; Web Link: http://nwb.slis.indiana.edu.
53. Huisman, M. and van Duijn, M. A. J., 2005, "Software for Statistical Analysis of Social Networks", in *Proceeding of the Sixth International Conference on Logic and Methodology*, Amsterdam, The Netherlands.
54. Barabási, A. L., 2002, *Linked: The New Science of Networks*, New York: Perseus.
55. Schaeffer, S. E., 2007, "Graph Clustering," *Computer Science Review*, **1**(1), pp. 27-64.
56. Romesburg, C. H., 2004, *Cluster Analysis for Researchers*, North Carolina: Lulu Press.
57. Faloutsos, M., Faloutsos, P., and Faloutsos, C., 1999, "On Power-Law Relationships of the Internet Topology", in *SIGCOMM '99: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, Cambridge, MA, pp. 251 - 262.
58. Krapivsky, P. L. and Redner, S., 2001, "Organization of Growing Random Networks," *Physical Review E*, **63**(6), pp. 066123.

Copyright © 2010 by ASME