Using Social Network Analysis for Mining Collaboration Data in a Defect Tracking System for Risk and Vulnerability Analysis

Ashish Sureka, Atul Goyal, Ayushi Rastogi

Indraprastha Institute of Information Technology, Delhi (IIIT-D), India

Presentation Outline

- Research Motivation
  Problem/Need encountered by practitioner
  Risk analysis in team structure - motivating application

- Research Aim
  To investigate social network analysis based techniques for mining defect tracking systems for risk analysis

- Related Work
  Literature survey of 20 papers on mining software repositories using social network analysis based techniques

- Research Contributions

- Research Methodology

- Solution Approach, Experimental Dataset

- Results

- Conclusions
Research Motivation

Teamwork, shared tasks, interaction, collaboration b/w team members is integral to SE/SD

For a manager or a team leader, an understanding of the following aspects are important:

- Team structure and topology
- Critical employees
- Core team
- Degree of centralization or de-centralization
- Sub-groups
- Leaders and communication bridges
- People with exclusive knowledge or skills

Motivating Application [Risk and Vulnerability]

- Employee attrition/turnover (common and inevitable)
- Unavailability of an expert (unforeseen circumstance)
- De-risk: critical employees, leaders, exclusive knowledge/skills

- Assessing/Identifying risk, intelligent and proactive decision making
- Can reduce loss and save time, important for project success
Research Motivation

Why team risk assessment is a *challenging*/arduous task

Constructing an explicit social or socio-technical network representation within an organization is a non-trivial task [Wolf2009]

Culture and process in an *OSS environment* is different than a CSS environment [Xu2006]

OSS projects
- Do not follow a pre-designed organizational structure [Bird2008]
- Usually dynamic, self-organizing, latent [Bird2008]
- Usually not explicitly stated [Bird2008]
- Lacks many of the traditional mechanisms (plans, system-level design, schedules, and defined processes) used to coordinate software development [Mockus2002]

Non-trivial to understand communities that build and support FLOSS software [Crowston2006]

There is no central control and planning in OSS [Madey2002]
Research Aim

**Broad Objective**

- To investigate techniques to support a practitioner (manager or project leader) in performing risk, threat and vulnerability analysis with respect to a software development team

**Specific Objective**

- To investigate social network analysis based techniques for mining software repositories (in particular bug tracking systems) for performing risk analysis

Defect tracking systems such as Bugzilla:

- Consists of online threaded discussion forum
- Developers discuss and collaborate with each other for problem solving
- Discussion archives contain wealth of information about people, knowledge team structure/topology, activity, interaction, expertise
**Related Work**

Literature *survey* of 20 papers on mining software repositories using social network analysis based techniques

SF=SourceForge, VA=Version Archive, SC=Source Code, ML=Mailing List, DTS=Defect Tracking

<table>
<thead>
<tr>
<th>Study</th>
<th>Repository</th>
<th>Purpose/Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Madey et al., 2002</td>
<td>SF</td>
<td>Testing power-law model in developer collaboration network at SourceForge</td>
</tr>
<tr>
<td>2 Luis et al., 2004</td>
<td>VA</td>
<td>Testing small-world phenomenon in committers and the modules networks</td>
</tr>
<tr>
<td>3 Cleidson et al., 2005</td>
<td>SC + VA</td>
<td>Study software artifacts and activities to uncover the structures of software projects Study collaborative work in large distributed groups such as open source Communities</td>
</tr>
<tr>
<td>4 Ducheneaut et al., 2005</td>
<td>ML + VA</td>
<td>Study relationships OSS newcomers develop over time with social and material aspects of a project</td>
</tr>
<tr>
<td>5 Huang et al., 2005</td>
<td>VA</td>
<td>Study grouping structures between developers and modules Verify Legitimate Peripheral Participation (LPP) process</td>
</tr>
<tr>
<td>Study</td>
<td>Repository</td>
<td>Purpose/Goal</td>
</tr>
<tr>
<td>-------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>6</td>
<td>Ohira et al., 2005</td>
<td>SF</td>
</tr>
</tbody>
</table>
| 7     | Bird et al., 2006 | ML + VA | Examine the relationship between communication and development  
Correlations between various social network status metrics and source code development |
| 8     | Crowston et al., 2006 | DTS | Empirically distinguishing core group of developers  
Study size and composition of the core groups |
| 9     | Howison et al., 2006 | DTS | Examine average centralization over time  
Study stability of participation in project communications |
| 10    | Sowe et al., 2006 | ML | Identification of knowledge experts in open source software projects  
Study the impact of knowledge brokers and their associated activities in open source projects |
| 11    | Xu et al., 2006 | SF | Testing scale-free property and small world phenomenon in OSS development community  
Study the effects of co-developers and active users in communication and information flow within the community |
<table>
<thead>
<tr>
<th>Study</th>
<th>Rep.</th>
<th>Purpose/Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Valetto et al., 2007</td>
<td>ML</td>
</tr>
<tr>
<td>13</td>
<td>Bird et al., 2008</td>
<td>ML</td>
</tr>
<tr>
<td>14</td>
<td>Martinez-Romo et al., 2008</td>
<td>VA</td>
</tr>
<tr>
<td>15</td>
<td>Meneely et al., 2008</td>
<td>VA</td>
</tr>
<tr>
<td>16</td>
<td>Pinzger et al., 2008</td>
<td>VA</td>
</tr>
<tr>
<td>17</td>
<td>Wiggins et al., 2008</td>
<td>DTS+ML</td>
</tr>
<tr>
<td>18</td>
<td>Sarma et al., 2009</td>
<td>DTS+VA+ML</td>
</tr>
<tr>
<td>19</td>
<td>Wolf et al., 2009</td>
<td>ML+VA</td>
</tr>
<tr>
<td>20</td>
<td>Meneely et al., 2010</td>
<td>DTS+VA</td>
</tr>
</tbody>
</table>
Research Contributions

- First Study on the application of *SNA techniques* to specifically study project *risk* and vulnerability (for an *OSS* environment) in *team-structure*

- The analysis of *defect tracking system* to study developer *collaboration and interaction data* is relatively unexplored. The only work that we have come across is by Crowston et al.

- First systematic *survey and classification of papers* (20 research papers categorized and listed in a chronological order from 2002 to 2010) focused on the topic of *mining software archives using social network analysis*. 
High-Level Solution Framework

Practitioners

Developer
Manager

Needs/problems [RISK ANALYSIS]

- Team structure and topology, Sub-groups
- Critical employees, Core team
- Degree of centralization or de-centralization
- Leaders and communication bridges
- People with exclusive knowledge or skills

Text Analytics | Data Mining | Social Network Analysis | Machine Learning

Knowledge Discovery and Pattern Recognition Techniques

Raw Data

Defect Tracking System | Discussion Archives | Version Control System | Source Code Repository

Software Repositories [Valuable Process Data, Historical Information]
Developer Collaboration Network

Mozilla BUG ID 250000

David: Bienvenu 2004-09-19 16:49:19 PDT

this patch also doesn't compile - there's no ';' at the end of the line. Are you sure you have assertions turned on? It needs to be a single assertion to get compiled.

Hans-Andreas Engel 2004-09-20 00:57:20 PDT

I temporarily added NS_ASSERTION, always hit this assertion, and checked that this assertion is hit indeed, indicating that debug build is needed.

> (...) there's no ';' at the end of the assertion. (...) Perhaps there was a mix-up in the applied patches?

In the revised patch (attachment 159437 [details]), there is
+ NS_ASSERTION(bytesToAdvance + (fLineOfTokens - fStartOfLine) + (int32) strlen(fCurrentLine), cannot advance beyond end of fLineOfTokens);

Thank you for running with this patch applied!

Dan Mosdale (:dmose) 2004-11-23 09:58:15 PST

Comment on attachment 159437 [details]
Remove strdup from AdvanceTokenizerStartingPoint; copy only of corrected assertions.

sr=dmosse


Fix landed by timeless: Bug 250000 Remove string copying in
|nsIMAPGenericParser::AdvanceTokenizerStartingPoint()|

Online discussion around a bug
Developer- Component Relationship

Bug 200001 - Issue when print the same report twice in the client[0002]

- **Status**: CLOSED FIXED
- **Product**: BIRT
- **Component**: Report Viewer
- **Version**: 2.2.0
- **Platform**: PC Windows XP
- **Importance**: P3 normal (vote)
- **Target Milestone**: 2.2.1
- **Assigned To**: Vincent Petry
- **CC List**: vpetry

See Also:

Component: AB
Assigned To: VP

Who (Developer) is working on What (Component)

Two Mode Network

Component-component relationship
Social network analysis programs/software

- Pajek: http://pajek.imfm.si/doku.php
  University of Ljubljana

- ORA http://www.casos.cs.cmu.edu/projects/ora/
  Carnegie Mellon University (CMU)

- UCINET: http://www.analytictech.com/ucinet/
  Analytic Technologies
Core, Semi-Periphery and Periphery Stratification/Characteristics

Information need of a manager

- Does the collaboration network follow a core-periphery pattern
- What is the size of the core, semi-periphery and periphery
- Who are the dominant, core, well-connected developers

Identification of core developers and peripheral developers is an important problem in software projects [Crowston2006]
Circular layout indicates: developers asa@mozilla.org and bzbarsky@mit.edu have high degree centrality.

Positioned at the center of the radial graph.

The Mozilla.org Staff Webpage lists the developer with the email id asa@mozilla.org as community quality advocate extraordinaire.

The developer with email id bzbarsky@mit.edu is listed on the Super-Review Policy webpage as one of the strong hackers enlisted by mozilla.org for universal code review coverage.
Two mode network: relationship between bug severity and developers
Authority Centrality and Betweenness Centrality

Top 20 developers in terms of authority centrality score

Derived from the developer-to-developer network

Ranking of top 20 developers based on centrality measures

Degree centrality: number of links/ties of a node

Betweenness centrality: intermediary, extent/degree to which a node lies between other nodes

Top 20 developers in terms of betweenness centrality score
Knowledge Exclusivity

Exclusivity, Knowledge: d2c

- benjamin@smedbergs.us
- ajschult@verizon.net
- sgautherie.bz@free.fr
- dougt@meer.net
- bzbarsky@mit.edu
- cbiesinger@gmx.at
- bienvenu@nventure.com
- asa@mozilla.org
- bugzilla@msmurphy.de
- martijn.martijn@gmail.com
- timeless@bemail.org
- brendan@mozilla.org
- bclary@bclary.com
- masayuki@d-toybox.com
- ratman@fastmail.fm
- dmik@innotek.de
- david@allouche.net
- VYV03354@nifty.ne.jp
- cstef@mail.ru
- emerson.pardo@gmail.com
Scatter plot between *authority centrality* (x-axis) and *betweenness centrality* (y-axis) score of developers.

Graph reveals that there are certain developers who have high betweenness centrality but not high authority centrality and vice-versa.
Identification of *fault-prone components*

Components which have *high density of critical bugs*

We observe that there are certain components which have received only critical bugs

The font size of the component labels are in the order of their frequency of bugs, i.e., the component Networking Cookies is having *maximum number of critical bugs*. 
- Identification of developers that have knowledge of varied components
- Developers who are expert in one or two specific components
- Number of developers working on each component
- The square nodes (representing components) are sized by degree.
- Figure shows that there are some developers working on only one component
- Figure shows developers working on many component
- bzbarsky@mit.edu.in has worked on maximum number of components (= 54 in this case)
- 84 developers have worked on the component Layout
A sub-group is a set of nodes who interact with each other more frequently or intensely than Nodes outside the group.

We study the dev-dev n/w to identify cohesive subgroups and teams/community

Degree partition of developers 556 nodes and 25 partitions based on the degree

9 developers of degree 10 which constitutes 1.61% of all the developers

There are 15, 14 and 18 developers with degrees 5, 6 and 7 respectively

There are 2, 1 and 1 developer with high degrees of 27, 29 and 30 respectively
Summary

We apply standard social network analysis approaches to investigate networks/relationships derived from an issue tracker

- Developer-Component Relationship
- Component-Severity Relationship
- Developer-Bug-Severity Relationship
- Developer-Developer Relationship

Derive hidden and interesting patterns useful to a project manager and developers

- Core, Semi-Periphery and Periphery Stratification
- Authority Centrality, Betweenness Centrality, Knowledge Exclusivity
- Clusters and Cohesive Subgroups

Present a systematic survey of the previous work in the area of social network analysis for mining software repositories.
Thank You