Motivation (1 of 3)

- We interviewed distributed developers, managers, and others (e.g., lawyers, HR personnel) in a Fortune 500 organization about their collaborative work practices.

- Without prompting participants, they frequently raised “trust” as a key concern for collaborations involving:
  - New collaborators (i.e., just met)
  - Previous collaborators (i.e., known for years)

Motivation (2 of 3)

- There are properties and attributes of teams and projects that influence the overall level of trust in collaborations
  - Antecedents of trust (Al-Ani and Redmiles 2009)

- Beyond that, one’s perception of others’ trustworthiness is based on expectations about others’ activities and behavior.

Outline

- Trust’s role in distributed software development
- Factors that influence trust
- Data that provides information about trust factors
- Building blocks for visualizations of that data
Motivation (3 of 3)

- Setting expectations of others in software development involves:
  - Recalling previous collaborations with them
  - Relying on recommendations from mutual colleagues
  - Remembering who was involved, how accessible they were, and whether they delivered on commitments

- This process is human-intensive

Unpacking Perceived Trustworthiness (1 of 2)

- “A belief or confidence about another party’s integrity (including reliability, predictability, and dependability) and benevolence (including goodwill, motives, intentions, and caring) in order to accept vulnerability.” (Mayer, Davis and Schoorman 1995; Ross and LaCroix 1996; Nooteboom, Berger, and Noorderhaven 1997)

Unpacking Perceived Trustworthiness (2 of 2)

- Two dimensions (McAllister 1995, Wilson et al. 2006)
  - Cognitive - competence and reliability
    - E.g., Submitting bug-free code
  - Affective - care and concern, “extra effort”
    - E.g., Notifying teammates before check-in conflicts occur

- Trust is about setting expectations

Scenario: Choosing a Team
After meeting with several graphic designers, lawyers, and finance managers last week, Mary, John, and Jeff meet to fill out developer positions for the project. At the end of the last meeting John volunteered to come up with a list of names of developers he knows to be skilled in visualization, an area all members agreed would play a critical role since the application will project renderings of high-resolution data sets with limited screen real estate. Mary is calling into the meeting from a developer conference in New York.

From his list, John suggests Sam from the algorithms group in Haifa, Israel. Saw his name mentioned in some e-mails flying around….looked at resume….10 years experience….

Mary: Algorithms Team Lead
John: Algorithms Senior Systems Engineer
Jeff: Networks Project Manager

Steve tells Jeff that Sam is working on several projects at the moment, and he’s playing a central role in two of them: he is designing the main APIs and web services for clients. Meanwhile, Mary looks at her e-mails and notices that Sam waited to implement many of the changes she needed in order to get her work done on time and rarely returned her e-mails on time.
The Decision: Scrap Sam

Mary: Algorithms
Team Lead

John: Algorithms
Senior Systems Engineer

Jeff: Networks
Project Manager

Cristy: Algorithms
Developer

Because of the innovative nature of the project and the risk involved, the managers and Mary decide not to pick Sam. His schedule is too busy and Mary is not impressed with his work habits and response rate—this contrasts with the high communication and prompt initiations and responses the project will require. In Sam’s place, Jeff recommends Cristy, a new developer, who comes highly recommended from his boss despite being in the Oregon-based “Algorithms” group for only 2 months.

Scenario Observations

• Assessing trustworthiness for effective collaborations is a human-intensive process
  ◦ Checking own documents and archives of collaboration history, finding out who was involved
  ◦ Traversing personal networks of colleagues
  ◦ Transitivity of trust
• Competence assumed a priori
  ◦ Shared work experience and reputation help inform decisions and set expectations

Other Scenarios

• Delivering on commitments
  ◦ Implementing code on which others depend (e.g. APIs)
  ◦ Fixing and closing out bugs
  ◦ Being transparent and accessible

• Being able to express half-baked or innovative ideas comfortably

Overarching Research Goal

• What if we had tools to support the process of setting expectations of trustworthiness by surfacing ‘trustful’ behaviors?
Ariadne is a tool that visualizes interdependencies, one type of collaborative trace.

We conducted a pilot experiment with six participants, mostly grad students.

We asked them about their perception of a project’s developers to help us investigate the question:

“Can visualizations of distributed collaborative traces influence a developer’s sense of other team members’ trustworthiness?”

First scenario (asking for help, identifying knowledgeable developers):

- Typically chose members they felt had the most interdependencies with others in the team.
- Indicated that the visualization and the representation of interdependencies in these visualizations led them to trust the central figure.
- Second scenario (sharing code):
  - Results were inconclusive; in general participants needed more information.
  - Some state less likely to risk relying on certain team member based on their inactivity.
  - Others stated that they would trust others with their code because “If someone breaks it, it probably wasn’t that great to begin with.”

The experiment provided evidence that visualizing collaborative traces can affect developers’ sense of trust toward others.

But participants indicated other evidence of developers’ activity would help them make trust judgments:

- Collaborative traces we didn’t happen to show.
Three Research Components

1. Investigate “collaborative traces” of trustworthy activity

2. Simulating collaborative traces that are characteristic of software development projects

3. Assess the impact of different visualizations of simulated collaborative traces on people’s sense of trust toward others

Data: Collaborative Traces

- Representations of past and current activity by a group of developers manipulating software development artifacts
  - The results of collaboration

- Examples:
  - Source-code change sets, call-graphs annotated with authorship, work item assignments and descriptions, e-mail and chat messages, etc.

Collaborative Traces Drive Awareness Tools (1 of 2)

<table>
<thead>
<tr>
<th>Task</th>
<th>Example Tools</th>
<th>Collaborative Traces</th>
<th>Input Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understand Change Management</td>
<td>ELVIN (Fitzpatrick et al. 2006), Command Console (O’Reilly et al. 2005), Source (German et al. 2006), Sub flirt (disk at 1992)</td>
<td>Change version logs and associated meta-data (e.g., commits, patches), file revisions, source-code call-graphs, bug reports, e-mail messages</td>
<td>CM Repository, E-mail Database, Bug Tracker</td>
</tr>
<tr>
<td>Evolution</td>
<td></td>
<td>Change version logs and associated meta-data (e.g., commits, patches), file revisions, source-code call-graphs, bug reports, e-mail messages</td>
<td>CM Repository, E-mail Database, Bug Tracker</td>
</tr>
<tr>
<td>Understood Developer Activities</td>
<td>FASTDash (Beitel et al. 2005), SHO (Ellis et al. 2007), Source</td>
<td>Change version logs and associated meta-data (e.g., commits, patches), file revisions, source-code call-graphs, bug reports, e-mail messages</td>
<td>CM Repository, E-mail Database, Bug Tracker</td>
</tr>
<tr>
<td>Find Relevant People and Files</td>
<td>ExpertiseRecommender (McDonald and Ackerman 2000), Answer Garden (Ackerman and Malone 1990), Hipikat (Mylyn) (Kersten and Murphy 2006), Team Tracks (DeLine et al. 2005), Source</td>
<td>Change version logs, source-code call-graphs, source-code authorship distribution, bug reports, e-mail messages</td>
<td>CM Repository, E-mail Database, Bug Tracker, Organization Chart</td>
</tr>
</tbody>
</table>

Collaborative Traces Drive Awareness Tools (2 of 2)

<table>
<thead>
<tr>
<th>Task</th>
<th>Example Tools</th>
<th>Collaborative Traces</th>
<th>Input Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avoid Conflicts</td>
<td>CollabVS (Dewan and Hegde 2007), TUKAN (Schummer and Haake 2001), Palantir (Sarma et al. 2003)</td>
<td>ChangesInProgress, ChangesCommitted, source-code call-graphs</td>
<td>CM Repository, Workspace, E-mail Database, Bug Tracker, GPS, Keyboard Input, Video, Shared Calendar, Workspaces</td>
</tr>
<tr>
<td>Determine Availability</td>
<td>Awarene (Begole et al. 2002), CommunityBar (Tee et al. 2006), Jazz (Hupfer et al. 2004)</td>
<td>Idle time, artifacts in current workspace, location of meetings in progress</td>
<td>GPS, Keyboard Input, Video, Shared Calendar, Workspace</td>
</tr>
</tbody>
</table>
Collaborative Traces for Trust (1 of 3)

<table>
<thead>
<tr>
<th>Trust Factors</th>
<th>Valence</th>
<th>Collaborative Traces</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring</td>
<td>+/−</td>
<td>Processes specifications, e-mail messages, instant messages</td>
<td>(Searle et al. 2007) (Al-Ani and Redmiles 2009)</td>
</tr>
<tr>
<td>Sharing Informative</td>
<td>+/−</td>
<td>Chat threads, e-mail messages, reading list postings, discussion forums postings, source messages</td>
<td>(Al-Ani 2009)</td>
</tr>
<tr>
<td>Leadership</td>
<td>+/−</td>
<td>Work item assignments, developers' interdependencies, source code authorship distribution, calendar</td>
<td>(Al-Ani and Redmiles 2009)</td>
</tr>
<tr>
<td>Rate</td>
<td>+/−</td>
<td>Organizational chart, source code authorship distribution</td>
<td>—</td>
</tr>
<tr>
<td>Same Location</td>
<td>+/−</td>
<td>Organizational chart</td>
<td>—</td>
</tr>
<tr>
<td>Project size</td>
<td>-</td>
<td>Design documents, readable specifications, project need for developers</td>
<td>—</td>
</tr>
<tr>
<td>Team size</td>
<td>-</td>
<td>Project description</td>
<td>—</td>
</tr>
</tbody>
</table>

Collaborative Traces for Trust (2 of 3)

| Project type | Project description | — |
|--------------|---------------------|——|
| Age          | Developer profiles, personnel database/employee directory | (Meadow and Iacono 1997) |
| Frequent Initiations and Responses | Chat threads, e-mail messages, reading list postings, discussion forums postings, work item status updates | (Jarvenpaa et al. 1999) (Searle et al. 2007) (Butler and Carroll 1994) |
| General Rating | Developers' interdependencies, e-mail messages, work item status updates, reading list postings, source code authorship distribution, calendar | — |
| Frequent Updates of Project Progress | E-mail messages, reading list postings, developers' interdependencies, work item status updates | — |
| Reputation    | Adjusted level systems, work item assignments, source code authorship distribution, developer profiles, calendar, organizational chart | (Ye et al. 2007) |

Collaborative Traces for Trust (3 of 3)

| Homophily      | +       | Developer profiles and resumes, personnel database/employee directory, organizational chart, work item assignments | (Fogg and Tang 1999) (Hertzum 2002) (Pyysninen 2003) |
| Team Diversity | -       | Organizational chart, work item assignments, source code authorship distribution, developers' profiles and resumes | (Pyysninen 2003) (Al-Ani and Redmiles 2009) |
| Multiple Responsibilities | - | Work item assignments, employees directory, team specifications, organizational charts | — |
| Use of Multiple Communication Media | + | E-mail messages, chat threads, reading list postings, discussion forums postings | (Chen, Bas et al. 2001) (Wierba, Grusz et al. 2003) |
| Shared Photographs | + | Developer profiles, employee directory | (Chen, Bas et al. 2002) |
| Expertise      | +       | Source code authorship distribution, developers' profiles, employees directory, expertise database, work item assignments and descriptions | (Gil and Aria 2006) |
| Availability   | +       | IP versus updates, developer specifications, mail calendars | (Liu et al. 2003) (Butler 1991) |
| Frequency of Meetings | + | Calendar | (Pyysninen 2003) |
Visual Form: Building Blocks

- A taxonomy of visualizations
- …that categorizes by family and parent visual representations (e.g. matrix on graph)

### A Taxonomy

<table>
<thead>
<tr>
<th>Family</th>
<th>Representation</th>
<th>Elements</th>
<th>Encoding Mechanisms</th>
<th>Layout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node-Edge</td>
<td>Graph, Icon</td>
<td>Nodes, Directed Edge, Un-directed Edge, Text</td>
<td>Color, Position, Area</td>
<td>Force directed, Tree</td>
</tr>
<tr>
<td>Bracket</td>
<td>Graph, Icon</td>
<td>Branches, Text, Links, Text labels</td>
<td>Color, Position, Area, Dependency Bands</td>
<td>Ordering on X-Y Plane</td>
</tr>
<tr>
<td>Matrix</td>
<td>Graph</td>
<td>Cells, Text</td>
<td>Color, Length</td>
<td>Force directed, Tree</td>
</tr>
<tr>
<td>Circle Packing</td>
<td>Tree diagram</td>
<td>Circles, Text label</td>
<td>Color, Area</td>
<td>Nested circles</td>
</tr>
<tr>
<td>Bracket</td>
<td>Tree diagram</td>
<td>Brackets, Text label</td>
<td>Color, Area, Angle</td>
<td>Nested concentric circles</td>
</tr>
<tr>
<td>Tree Diagram</td>
<td>Tree diagram</td>
<td>Trees, Text label</td>
<td>Color, Area</td>
<td>Nested rectangles</td>
</tr>
<tr>
<td>Indented Text</td>
<td>Tree diagram</td>
<td>Text</td>
<td>Position</td>
<td>Nested text</td>
</tr>
<tr>
<td>Stacked Area Chart</td>
<td>Chart</td>
<td>Area, Text labels, Text length, Area, Color, Angle, Shape</td>
<td>Ordering on X-Y plane</td>
<td></td>
</tr>
<tr>
<td>Bar Chart</td>
<td>Chart</td>
<td>Text, Text labels, Text length, Area, Color, Angle, Shape</td>
<td>Ordering on X-Y plane</td>
<td></td>
</tr>
<tr>
<td>Spreadsheet</td>
<td>Chart</td>
<td>Text</td>
<td>Color</td>
<td>Ordering on X-Y plane</td>
</tr>
<tr>
<td>Trust Spheres</td>
<td>Chart, Tree diagram</td>
<td>Text, Labels, Text, Area, Length, Position, Area, Angle, Shape</td>
<td>Markups</td>
<td></td>
</tr>
</tbody>
</table>

### Mapping Visualizations to Trust Factors

<table>
<thead>
<tr>
<th>Family</th>
<th>Node-Edge</th>
<th>Bracket</th>
<th>Matrix</th>
<th>Circle Packing</th>
<th>Stacked Area Chart</th>
<th>Bar Chart</th>
<th>Spreadsheet</th>
<th>Trust Spheres</th>
<th></th>
<th>4/19/11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiation</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reputation</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authority</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leadership</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Years Experience</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication Media</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Homophily</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shared Information</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shared Photographs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency of Meetings</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Team Diversity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Updates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Team Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitoring</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trust Spheres</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4/19/11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4/19/11
As node-edge visualizations get highly connected, readability suffers:
- Line crossings
- Overlapping nodes

Layout algorithms position nodes and edges differently each time.
Notes

- Line crossings are impossible
- Symmetry or asymmetry can be seen across the diagonal
- Patterns are much more clear than node-edge visualizations
- Frequently used to describe relationships
  - Communication
  - Dependence

Ordering Matters

"Bad" Ordering

"Good" Ordering

CirclePacking

Managers
Team Leads
Developers
Testers
Graphic Designers
Notes

- CircePacking diagrams show containment
- Attributes can be compared using area judgments
- Does not use space efficiently but wasted space essentially reveals the hierarchy
- Can be used to group similar nodes
  - Role
  - Skills

Notes

- Recursively dividing rectangles makes efficient use of space
- Area is used to quantify attributes
Similar advantages of treemap and circle packing
- Reveals hierarchy like circle packing and is as space-filling as treemap
- Require much vertical space
- Allows rapid scanning of labels and multivariate data can be shown adjacent to the hierarchy
Stacked Area Charts

- Stacked area charts show cumulative total and relationships of parts to the whole over time.
- E.g., show the total number of messages sent by a developer over time, broken down by imitations and responses.

Bar Charts and Spreadsheets

Mashup
Lightweight vs. Monolithic Visualizations

- E.g., small “app”-like visualizations vs. mashups
- Textual decorators
- Can impact adoption of the visualizations

Other Visualizations

- Pie charts
  - Percentage of bugs fixed
- Map-based visualizations
  - Sense of availability due to geographic distance and time zones
There are many types of visualizations. They can be roughly categorized into a taxonomy based on the task:
- Show relationships
- Show structure
- Show changes over time
- Facilitate comparisons

When designing visualizations, keep in mind the data and task:
- Collaborative traces/trust

Designing visualizations is a craft. We would like a lightweight way to cull and refine visualizations.

Inspection Methods:
- Cognitive Dimensions of Notations (Green and Petre 1996)
- Cognitive Walkthrough (Wharton, Reiman, Lewis et al. 1994)
- Heuristic Evaluation (Nielsen 1994)
- Physics of Notations (Moody 2008)

“framework for describing the usability of notational systems…and information artefacts”

Used by many researchers (Pereira et al. 2008, Li et al. 2007, Tukiainen 2001; Cox 2000)

Provide surface analysis, not extensive analysis (lightweight)

Questions?
- Thank you!
Example Dimensions

- **Viscosity**
  - Indicates how much effort is needed to make a change in the notation, such as the layout

- **Diffuseness**
  - Refers to the number of symbols required to express a certain result or meaning

- **Hidden Dependencies**
  - Refers to whether relationships between entities are visible

- **Closeness of Mapping**
  - Indicates how closely the notations corresponds to the problem world

- **Hard Mental Operations**
  - Refers to how much mental processing is required at the notational level rather than at the semantic level

Examples of notations:

- (Moody 2008)
- New design theory
- “Synthesized from theory and empirical evidence…of how visual notations communicate”
- Consists of principles for designing effective visual notations

Example Principles

- **Semiotic Clarity**
  - There should be a 1:1 correspondence between semantic constructs and graphical symbols

- **Perceptual Discriminability**
  - Different symbols should be clearly distinguishable from each other

- **Semiotic Transparency**
  - Use visual representations whose appearance suggests their meaning

- **Dual Coding**
  - Use text to complement graphics

- **Cognitive Fit**
  - Use different visual dialects for different tasks and audiences