 Foundations for the Design of Visualizations that Support Trust in Distributed Teams

Erik H. Trainer
Department of Informatics
University of California, Irvine
Irvine, CA, 92697-3425, USA
+1-415-658-5933
etrainer@ics.uci.edu

David F. Redmiles
Department of Informatics
University of California, Irvine
Irvine, CA, 92697-3425, USA
+1-949-824-2901
redmiles@ics.uci.edu

ABSTRACT
We seek to provide design principles for software tools intended to support the development of trust in distributed teams. As such, we present a “design space” for such tools that consists of three elements: trust factors, collaborative traces, and visual representations. Trust factors are aspects of work shown in the research literature to influence one’s perceived trustworthiness of their team members. Collaborative traces are representations of past and current work done by team members manipulating project artifacts. Collaborative traces provide information about a trust factor. Visual representations consist of a set of visual abstractions of the collaborative traces arranged in a layout that provides users with the ability to formulate accurate perceptions of their team members’ trustworthiness with respect to a particular trust factor. Because the eventual goal of this research is to produce visual interfaces that support the development of trust, we apply elements from the design space to the design of three example visualizations. We conclude by assessing the value of the model and outlining future work.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces – graphical user interfaces (gui), prototyping, theory and methods. H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces – asynchronous interaction, theory and models, computer-supported cooperative work. K.4.3 [Computers and Society]: Organizational Impact – computer-supported collaborative work.

General Terms
Design, Human Factors, Theory.

Keywords
Visual Interface Design, Interactive and Collaborative Interfaces, Trust, Distributed Teams.

1. INTRODUCTION
Trust is a significant human-oriented aspect of successful and productive collaborations [12][25]. Trust is often defined in terms of one person’s expectations of another. For example, Furst et al. found that an individual’s trust in their team refers to the likelihood that team members will live up to expectations [10]. Likewise, Sabherwal considers trust a “state involving confident positive expectations about another’s motives with respect to oneself in situations entailing risk” [17].

A lack of information about distributed colleagues can work against developing trust in distributed teams. Remote workers are likely to have much less information and lower quality information about their remote partners [7]. For example, if Chris had known Alex was working on three projects, the past two months of which he was in Brazil working on fixing bugs, he could have modified his expectations with respect to his availability. Cramton observed that the lack of situational knowledge and the reduced ability to process it effectively can cause individuals in remote teams to attribute breakdowns to the individual rather than the situation itself, eroding team cohesion and lasting solutions [7]. In this instance, a loss of trust is more difficult to repair. When what is actually observed in the world (e.g., a developer’s failure to deliver work on time) clashes with expectations (e.g., their perceived ability to deliver on time), trust can be slow to build between collaborators.

There are several aspects of the process of setting expectations in the context of globally distributed software development that can be addressed and improved. First, developing a sense of trust can take time due to the lack of information about other colleagues’ activities and the lack of ways to manage this information in order to set realistic expectations. Distributed developers can ultimately reach the same levels of trust as colocated team members; it just takes longer [25]. A tool that can partially automate this process can provide time savings. Second, in software development, the interactions that help set expectations are typically hidden in project repositories or incomplete documents over time. Further, a lack of situational information about colleagues can negatively and more importantly, inaccurately, bias trust judgments. If a developer is involved in multiple projects across several time zones, one shouldn’t expect same day responses, for example. An approach that renders this information explicitly from project and team artifacts can prepare people to make trust judgments based on meaningful and more complete data. Third, the sheer number of artifacts involved makes this data, if presented in textual form, difficult to interpret. Visualizations can be more effective in revealing and summarizing this information.

Therefore, we seek to lay the foundation for designing visual interfaces that support the development of trust in distributed teams. In Section 2, we define the research problem and motivate our proposed solution. In Section 3, we begin laying the foundation for the design of visual interfaces to support trust. In Section 4, we present our model of the design space for such tools, and provide some example visualizations designed from
that space in Section 5. We discuss the value and limitations of the model in Section 6 and conclude in Section 7.

2. RESEARCH APPROACH

Given the motivational factors in the introduction, we formulate the following, overarching research question: Can a software tool usefully provide information that engenders trustworthiness among distributed team members? Before that question can be answered, however, one must know what “information” such a tool would provide, as well as from where it could be acquired.

2.1 Collaborative Traces

Software design and development activities typically produce a project memory of archival artifacts such as source-code, e-mail lists, design documentation, bug reports and change histories of all this information that developers use as a means toward coordinating the smooth flow of work. For example, dependency call-graphs annotated with authorship information at the line level for each artifact can be acquired from a project’s Configuration Management (CM) repository. This information can help developers assess the impact of changes to source-code and whether they will need to coordinate with other developers as a result [19].

Software engineering researchers have created and evaluated a host of awareness tools. In earlier work [21], we surveyed 40+ different visualization tools in the research areas of: software tools and environments, empirical software engineering, computer-supported cooperative work, awareness, and visualization (e.g., at the ICSE, CSCW, ECSCW, SOFTVIS, AVI, and VL/HCC venues). Common goals of these tools include providing information about module dependencies and real-time developer activity to proactively avoid check-in conflicts and “breaking the code,” understanding change management and evolution of code, recommending the right people and the right artifacts at the right time to match the task at-hand, and determining opportunistic times to contact developers (and the latter’s willingness, in turn) for help, all of which change over the course of development. Research has found that not providing this information can increase the chance of behavioral invisibility and the possibility of reducing trust [25].

From the perspective of design, one important property of these tools is that they visualize information from collaborative traces [22]. Collaborative traces are representations of past and current activity by a group of developers manipulating software development artifacts. For example, collaborative traces include source-code change sets, source-code call graphs and interdependencies annotated with authorship information, work items (such as bugs and feature requests), and e-mail and chat messages. They are generated by collaborative events such as writing source-code, fixing bugs, and correspondence. Table 1 lists example collaborative traces organized by the tasks they support and well-known research tools that utilize them, as identified in previous work.

2.2 Collaborative Traces for Trust

Visualizations of collaborative traces provide information that increases a developer’s sense of awareness about developers and the artifacts with which they work. Answering the research question proposed at the beginning of Section 2 requires understanding which of these traces can provide information about others’ trustworthiness. Table 2 is a matrix that serves this purpose. The columns of the matrix are trust factors, i.e., information that affects people’s perceptions of others’ trustworthiness. The authors acquired these factors from a review of the literature on trustworthiness. Rows in the matrix are collaborative traces as well as other data that, although do not fit the definition of a collaborative trace because they are not an outcome of work, provide information about an individual or the organization, such as an org. chart or the time zone in which a developer is located.

For example, change-sets and authorship from source-code can reveal information about developers’ expertise and their general development activity. A developer whose code is used by everyone on the team can be perceived as having expertise in that area of the codebase [22]. In addition, the rate at which developers send out and respond to e-mails and instant messages can signal their overall willingness to initiate and respond to the cares and concerns of others. Developers located in different time zones but who are quicker to respond to e-mails than developers in the same time zone may be perceived as especially trustworthy. Developers who resolve other developers’ work items may be perceived as particularly benevolent and as a consequence, gain their colleagues’ affective trust. Lastly, developers who use multiple communication media: are available by chat or are active in the project’s mailing list rather than just e-mail alone, may be perceived by some to be particularly trustworthy.

2.3 Visual Representations for Trust

Visual representations summarize the information provided by collaborative traces. Initial research in this area has shown that visual representations of traces such as dependencies created from source-code and change-sets can influence developers’ perceptions of their team members’ trustworthiness [22].

Perhaps the fundamental task in presenting information visually is deciding which visual representations to use. Nowadays, guidelines for visualizing data are available to developers through web-based applications (e.g. ManyEyes, Swivel, Google Chart Tools), and typically include a baseline set of visual representations organized by tasks they support. Examples include node-edge diagrams and matrices for analyzing relationships between entities, bar charts for comparing values, circle packing to examine hierarchical relationships, and scatterplots to determine the relationship between two variables.

Prior to the emergence of web-based advice, researchers sought to understand the underlying perceptual reasons behind the suitability of different kinds of visual representations for data, thereby lifting the level of discussion regarding the design of visual interfaces [4][15]. While the aforementioned web-based tools have succeeded in making the high-level advice more accessible, they have not lifted the level of discussion. In contrast, our work seeks to lift the level of discussion by providing advice on the design of visual interfaces to support trust. Table 3 is a matrix of visual representations (rows) and collaborative traces (columns) that provides advice for looking at a trace and seeing data-wise, whether it can be potentially represented in a particular visual form.

For example, by flattening the hierarchy elements in Circle Packing [24] visual representations, developers can be grouped by their similarities, such as their areas of expertise in a particular field, or their location. Identifying this homophily can influence developers’ sense of trust toward one another. Information about expertise can be obtained from collaborative traces such as source-code and change-sets, as well as other data such as org. charts and personnel profiles.
Table 1. Collaborative traces and other data (rows) mapped to trust factors identified in the literature (column).

<table>
<thead>
<tr>
<th></th>
<th>Initiations and Response</th>
<th>Same Location</th>
<th>Role</th>
<th>Expertise</th>
<th>Reputation</th>
<th>Availability</th>
<th>Leadership</th>
<th>Years Experience</th>
<th>Communication on Media</th>
<th>Homophily</th>
<th>Shared Information</th>
<th>Shared Photographs</th>
<th>Frequency of Meetings</th>
<th>Team Diversity</th>
<th>Project Updates</th>
<th>Team Size</th>
<th>Project Size/Type</th>
<th>Monitoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project descriptions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chat thread</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instant messages</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E-mail messages</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mailing list postings</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calendar</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Keyboard input</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time zone</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Personnel profiles</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work items</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td>X X</td>
<td>X X X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source-code</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Org. chart</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change sets</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Visual representations (rows) mapped to trust factors identified in the literature (column).

<table>
<thead>
<tr>
<th></th>
<th>Initiations and Response</th>
<th>Same Location</th>
<th>Role</th>
<th>Expertise</th>
<th>Reputation</th>
<th>Availability</th>
<th>Leadership</th>
<th>Years Experience</th>
<th>Communication on Media</th>
<th>Homophily</th>
<th>Shared Information</th>
<th>Shared Photographs</th>
<th>Frequency of Meetings</th>
<th>Team Diversity</th>
<th>Project Updates</th>
<th>Team Size</th>
<th>Project Size/Type</th>
<th>Monitoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node-edge</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Matrix</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scatterplot</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CirclePacking</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sunburst</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Treemap</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indented text</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Chart</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bar Chart</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spreadsheet</td>
<td></td>
<td>X X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map</td>
<td></td>
<td>X X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line-based</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
That is, to support the development of trust, tool designers should support trustworthiness between distributed team members. We propose the following model for the design space of tools to support trustworthiness between distributed team members:

Model of Design Space =

{ Trust factors, Visual representations, Collaborative traces }

That is, to support the development of trust, tool designers should pick trust factors they would like to maximize (e.g., responsiveness, expertise, or location from the columns in Table 1). They should then choose visual representations for those factors (e.g., bar charts, spreadsheets, or maps from Table 2), and collaborative traces and other data (e.g., source-code, work items, or time zones from the rows in Table 1 and columns in Table 3) that provide information about the trust factors. The intersection of all three forms a triple in 3-D space (Figure 1). We are careful to acknowledge that data on each axis are not continuous; the purpose of the axis model is to illustrate the design space only.

Thus, each point in the space represents a particular combination of traces, trust factors, and visual representations. Because multiple collaborative traces can be used for a single trust factor (Table 1), points along the collaborative trace axis represent a combination of traces, where the order of traces does not matter. Thus {e-mail, instant message} is the same as {instant message, e-mail}. An example triple in the space could be ({availability},{e-mail, instant message},{bar chart}). This point is distinct from ({availability},{e-mail},{bar chart}) but the same as ({availability},{instant message, e-mail},{bar chart}).

### 4. EXAMPLE APPLICATIONS

In this section, we demonstrate how the model of “design space” might be used to design several visualizations to support trust in distributed development teams. In the example below, we use

<table>
<thead>
<tr>
<th>Table 3. Visual representations (rows) mapped to collaborative traces and other data (column).</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Table 3" /></td>
</tr>
</tbody>
</table>

Additionally, circles can show all developers who have been assigned to or resolved the same work item, a collaborative trace representing the description of work to be completed.

Circles can pack developers by location as well. Location data can be derived from developers’ work sites, personnel profiles, and time zones. In bar chart visualizations, source-code could be used as a collaborative trace to plot the number of modules authored per developer in a project. The length of bars can be rapidly compared to make quick judgments about who contributes in the project. Alternatively, other information that influences trustworthiness such as the developer’s years of experience, or the developer’s daily response rate to e-mail or instant messages could be shown using length of bar as the encoding mechanism. The number of work items resolved per developer could also be easily shown with a bar chart.

Understanding which trust factors to present, how to present them in a visual interface, and where the data will come from is a necessary step in designing tools to support the development of trust. In the next section, we present a model that embodies these requirements.

### 3. MODEL OF DESIGN SPACE

In Section 2 we used observations from the literature on software engineering and awareness in concert with literature on trustworthiness to provide insights into what information would need to be acquired in order to develop a tool that usefully engenders trustworthiness among distributed team members. This information, along with appropriate visual representations of it, infuses our model of the design space for such tools. In the next section, we present our model as well as example applications of the model to visual interface design.

#### 3.1 The Model

We propose the following model for the design space of tools to support trustworthiness between distributed team members:

Model of Design Space =

{ Trust factors, Visual representations, Collaborative traces }

of all three forms a triple in 3-D space (Figure 1). We are careful to acknowledge that data on each axis are not continuous; the purpose of the axis model is to illustrate the design space only.

Thus, each point in the space represents a particular combination of traces, trust factors, and visual representations. Because multiple collaborative traces can be used for a single trust factor (Table 1), points along the collaborative trace axis represent a combination of traces, where the order of traces does not matter. Thus {e-mail, instant message} is the same as {instant message, e-mail}. An example triple in the space could be ({availability},{e-mail, instant message},{bar chart}). This point is distinct from ({availability},{e-mail},{bar chart}) but the same as ({availability},{instant message, e-mail},{bar chart}).

![Figure 1. Axes of the “design space” of visualizations that support trust.](image)
availability as the trust factor of interest and show which collaborative traces and visualizations should be used as a result of that selection.

One’s availability has been shown to influence others’ perceived trustworthiness of them [1][6][12]. In general there are two aspects of availability: accessibility and responsiveness [2]. A user study by McDonald and Ackerman revealed that, all other things equal, employees prefer to connect with people who are physically or organizationally close [16]. These preferences imply two aspects of accessibility. The first is practical accessibility, which refers to the individual’s work unit location and their workload (e.g., number of parallel projects). The second aspect of accessibility is organizational accessibility, which refers to their position in the hierarchy and job title and description (their role in the organization). Certainly, physical location has a significant influence on the effectiveness of collaboration. For example, consider person A and B in Irvine, person C in New York, and person D in Israel. For A, the availability <A, B> is much greater than <A, C> and <A, D>. Literature has suggested that there may be no big difference between <A, C> and <A, D> although the physical distances varies greatly [5].

The notion of responsiveness, or willingness to respond, relates to a person’s accessibility. Studies of real-world email datasets, such as the ENRON corpus, have concluded that the average corporate reply time is about 1 day. One quantitative study of the ENRON data found that the average response time of over 16,000 e-mails was 28.8 hours. The researchers found that a typical user responded to 84% of messages within a day and 95% of messages within 5 days [13]. Another quantitative study analyzed 315 million e-mails over a 3-month period and found that the median reply time to a message was approximately 1 day [14]. Qualitative studies have found that individuals expect others to reply to them within 1 day. When questioned about their reply times, participants maintained informal “Responsiveness Images,” which projected to others that they were likely to reply within the work day or 24 hours of receiving a message [20].

From Table 1, the collaborative traces that correspond to availability are email messages, instant messages, and mailing list postings. Other data include organizational charts and work unit site locations from project descriptions and time zones.

To demonstrate the applicability of our model, we have designed three potential visualizations in the intersection of the availability trust factor and the collaborative traces outlined in Section 4.2.

4.1 Availability Radar

The first visualization we created using the design space is the “Availability Radar” visualization (Figure 2). The “Availability Radar” groups developers by their proximity to the user of the visualization. However, the groupings are not defined by physical location alone. They take into account account time zone as well, which is just as important. As such, the groupings are defined as follows:

- Green: Same building
- Yellow: Same time zone
- Red: Different time zone

The further someone is physically from the user’s current location, the farther out they are placed along the X and Y axis from the center of the visualization. A black “X” in the visualization represents a manager whereas a white “X” in the visualization indicates a non-manager. We use color to encode role in the organization because of evidence from our data on trust that indicates that people perceive managers to be less available in hierarchical organizations. “Availability Radar” was inspired by the design of the CirclePacking visual representation and modified to show “levels” of availability without a strict hierarchy. The circles are used to group “children” nodes, i.e. developers together that have the same properties, i.e. same location. The collaborative traces that provide the data needed for the visualization are organizational charts and time zones.

![Figure 2. A team in which the closest managers are located in the same time zone, thus the user of this visualization should trust them to be available to communicate.](image)

One way to make this visualization more malleable and potentially more useful, would be to choose other dimensions for the coloring such as overlap in the time of day. We address this possibility in Section 4.3.

4.2 Responsiveness Bars

The second visualization (Figure 3) shows developer’s responsiveness, the second aspect of availability. The insights about levels of responsiveness described previously in Section 4 were the basis from which the categories of “same,” “next” and “within 5” days were designed. The bar chart visualization was chosen because it allows for quick comparisons between numerical data: the user in question’s response rate and the average reply rate. We see in the example above that the contact’s response rate is slightly below average. The collaborative trace that provides this visualization with data is e-mail. One could imagine the same visualization using instant messages or mailing list postings.

![Figure 3. This contact (blue) is likely to respond to 7 out of 10 e-mails during the same work day, and almost all e-mails within 5 days.](image)

4.3 Responsiveness + Time Zone Overlap

While the availability radar looks at just distance and time zone as measures, overlapping time in the day between locations can be more useful. The third and final visualization example (Figure 5) is an alternative to the previous visualization. It is also...
designed to show responsiveness, but shows individuals’ average reply time to e-mail relative to the mean reply time of all individuals. For example, suppose person A replied to 10 emails in the following time interval (1, 2, 1, 4, 2, 9, 23, 11, 8, 3) <in hours>. Their average reply time is 6.4 hours. After calculating every contact’s average reply time, we standardize them according to their distribution. We then calculate standard scores for each individual’s average reply time and plot them along a simple linear scale to show how far, to the left or right (below or above), the scores are from the mean average reply time. Stacking the horizontal axes on top of one another for each contact searched affords quick comparisons between the position of each contact’s standard score to determine which contact is more responsive to e-mails.

![Figure 4. The bottom contact responds to e-mail more quickly than the top, despite being 13 hours ahead of him.](image)

In distributed teams, an individual’s reply time to a message is likely to be influenced by time available in the day for work. What is 1 pm over the course of a normal work day for one developer may be 3 am for another developer. As such, one should not expect a response within the same work day. Using this insight, we augment the reply time visualization with a visualization of the time zone overlap between the user and the contact in question. The left half of the visualization consists of rows corresponding to each hour in the day for the user, while the right half shows all hours in the day for the contact in question. A green line indicates normal working hours (8 am–5 pm). A yellow line indicates hours during the day where the individual is likely to be active, but not necessarily at work (7 am and 6 pm–9 pm). A red line indicates normal sleeping hours (10 pm–6 am). Overlaps (and the lack of overlap) in working and non-working hours are immediately obvious given this color coding.

This visualization was inspired by line-based visualizations, such as SeeSoft [8]. Although a map visualization could also show time zone information (Table 2), it was not chosen for two reasons. First, there is not enough screen real-estate to show the entire map. Second, Map-based visualizations typically indicate differences in time zone by shading dark for night time and light for day time. The beginning of one person’s work day may be the end of another’s. By breaking down the time zone overlap by hour, users of the visualization can intuit good times to meet and follow-up on correspondence. Each line is color-coded by the time of the day in the particular region. When a single row is viewed, it is obvious whether the two times belong to the same time group. In this design, the cells are “squashed” to the point where each row is simply a line because what is necessary to show is the overlap, not the reading of a particular numerical value, i.e. the exact time. The collaborative traces required are e-mail and time zone. Like the previous example, one could imagine instant messages and mailing list postings in addition to e-mail.

In addition to showing overlap in time zone, this visualization could be extended by coloring the overlap in working hours. That is, it is entirely possible that a distributed team member may be working other hours than the standard 8–5. In the style of other visualizations (e.g., [3]), we could provide an alternate coloring that shows what hours of the day team members’ work activities are likely to be in alignment. Taken together with time zone overlap, such a feature could provide a useful precision regarding one’s expectations of their remote team members’ behavior.

5. DISCUSSION
In this section, we discuss the value and limitations of the model, as well as foreshadow our future work in this area.

5.1 Assessing the Model
We argue the model presented here has value for three primary reasons. These include the following: organizing concepts from the literature, accommodating diverse tools, and providing insight into visual interface design.

A thorough review of the literature on trustworthiness was performed to identify the factors that influence developers’ perceptions of their team members’ trustworthiness. The review was performed across multiple domains: organizational science, management, global software development, and business and e-commerce. Forward citation searches of studies on the definition of trustworthiness were conducted to identify models of trustworthiness constructed based on longitudinal studies of distributed teams. Backward citation searches of these empirically validated models were conducted to identify definitions of trust not revealed by the original searches.

The model helps us think about the design space of visualizations to support trust in a systematic way. One way to think about the space is in terms of existing tools. Figure 5 shows a subset of these tools surveyed in previous research [21]. The important thing is not showing precisely where the tools fall in our design space, but to illustrate the space itself.

As an example, the first visualization example in Section 4.3 lies at the intersection of the availability trust factor, organizational chart and time zone collaborative traces, and circlepacking visual representation. The second visualization, “Responsiveness Bars,” used the availability trust factor, email, instant messages, and mailing list postings. It uses the bar chart visual representation. The third example, “Standardized Reply + Time Zone Overlay” used the availability trust factor, email, time zone, instant messages, and mailing list postings. It used two visual representations: linear scale and line-based. Awareness, a tool also designed to show the availability of developers [3], fits into the three-dimensional space. It uses keyboard input and calendar information as its collaborative traces and an actogram as the visual representation. Other tools surveyed in previous work [21] include Ariadne versions 1 [19] and 2 [20], Tesseract [18],
SoftCHANGE [11], ELVIN [9], and SeeSoft [8]. This set of existing tools fits into the design space of the model.

Laying out tools in this way raises the question of whether positioning visualizations within the tools rather than the tools themselves will yield more useful guidance for designing visual interfaces. If true, we would be able to take “good” visualizations from different tools and apply them to new designs. A single visualization that can address multiple traces and multiple trust factors may be more desirable than multiple visualizations that address the same set of factors. For example, the time zone overlap visualization may be able to be extended to encode more traces, such as calendars and e-mail messages, using different visual representations such as a color gradient. This approach effectively takes visual representations from both the SeeSoft and Awarenex tools. In this way, discussing the visualization as the unit of analysis may provide more insight to design than considering each tool in turn.

Additionally, we might emphasize some attributes (points on each axis) over others, depending on the context. The model provides value by giving us different choices for different situations. For example, when assigning team members to particularly innovative software projects, managers may value expertise over availability, despite the fact that that expertise is distributed throughout many time zones. Unless the manager is a software developer, it is unlikely that visualizations of collaborative traces such as work-items and source-code will be meaningful. Rather, personnel profiles and org. chart information will be more appropriate.

On the other hand, developers who are being “onboarded” to such a project will likely need to quickly learn the inner-workings of a system as well as learn who to go to for help. This requires knowing about their colleagues’ expertise. As such, collaborative traces like source-code and work items more meaningful because these are the artifacts with which they most often work. Alternatively, availability may be very important to remote developers who must coordinate their changes with one another (e.g. they share highly interdependent code), but whose differences in time prevent ideal times to meet one another. In the absence of large shared displays, compact spreadsheet line-oriented visualizations like the “Time Zone Overlap” can indicate good times to meet one another “live” over the phone.

The model also gives insight into the visual interface design process by showing that information about the various trust factors can be automatically extracted from existing projects. A fundamental question of visual interface design is from where the data will be acquired and how accurate it will be. By operationalizing trust in terms of collaborative traces, the model makes access to real-world data from multiple projects feasible without the worry that insufficient or incomplete data will be collected. Moreover, automatically extracting traces eliminates the need for time consuming surveys, field studies, and interviews to get at the same data.

5.2 Limitations and Future Work

There are two significant limitations to the value of the model described here. First, we should be careful not to reduce trustworthiness to a matter of collaborative traces. Our model characterizes attributes of the trustee, but perceived trustworthiness is strongly psychologically motivated as well. Propensity to trust is a general personality trait of the trustee that conveys a general expectation of how trusting one is [12]. Propensity can influence the extent to which one trusts another person before data on the latter is available. Indeed, we noted in our interviews with distributed team members that it is common for trustors to adjust their initial sense of trust toward others as certain information about them comes to light. Even individuals who described themselves as non-trusting in nature became more trusting of others after having worked with them. Having this information sooner than later can act as a correction mechanism early in the collaboration. It might be interesting in future work to explore the influence of propensity to trust on subjects’ perceptions of distributed team members’ trustworthiness.

Second, the design space as proposed by our model is very large. Yet our model helps navigate this space; in particular, tables 1, 2, and 3 substantially restrict this overall space. As we build, evaluate, refine, and cull new visualizations using these tables we can not only add new data points to the space but scope to the space to “useful” visual interfaces. At current, one limitation of the model is that it does not take context into account. As a first step, we are interested in considering “task” and “expectations” as other dimensions in the model that could be subsumed under “context.” For example, the responsiveness visualizations such as time zone overlap may be useful when developers are composing an e-mail to colleagues in remote locations. Thus the task would be sending an e-mail. An expectation of responsiveness could be based on the mean across all team members, or the mean reply time for the particular time zone in which the team member is located. Current evidence [14][23] suggests that one should expect a reply in about one day, but surely this will depend on where that team member is located. In our future laboratory studies we hope to investigate responsiveness expectations when members of the team are globally distributed.

6. CONCLUSION

This work was motivated by the research question of whether software tools can support the development of trust in distributed software development teams. In order to assess the feasibility of these tools, however, principles by which they should be designed are needed. This paper lays the foundation for expressing such principles by building up a “design space” for visualizations that support the development of trust. The design space consists of three elements: trust factors, collaborative traces that provide information about those factors, and visual representations of the traces. We discussed the elements of the design space in the context of awareness tools and developers’ sense of trust toward others and provided illustrative examples. We concluded by discussing the value and limitations of our model. Our future work will include additional visual prototypes as well as an
investigation of which attributes in the space should be emphasized over others depending on the context.

7. ACKNOWLEDGMENTS

This work is supported by the U.S. National Science Foundation under grants 0943262, 0808783 and 1111446. We thank our colleagues Oliver Wang and Matthew Bietz for reviewing early drafts of this paper.

8. REFERENCES


