Project Integer[1] General User Interface Objectives and Requirements

Version	Date	Description
1.0	12/2/2013	Initial version.

Introduction

This document outlines general objectives and requirements for the Integer human interface. It is not intended to be a substitute for specific requirements found in functional requirements documents that may be developed from time to time. This background is intended to help the engineering effort in a number of areas:

- Determination of client and server technologies that will best meet these objectives.
- Assist in the development and use of APIs between the client(s) and servers that comprise Integer.
- Lay a foundation for the types of interactions anticipated between different users and the system that can be used to define and implement functions.
- Identify the types of users, roles and their characteristics that will guide how different functions might be realized with the interfaces.
- Assist in the prioritization of which functions belong in different types of clients.
- Clarify what is required for different client types.

Many of the issues raised in this document are not resolved but are left for further research and review. The document places some initial stakes in the ground for discussion purposes.

Integer as an Engineering and Operational System

Integer is an engineering application, the majority of the users that access the system are expected to be experienced and trained in their respective fields such as software engineering, network or server administration and application operations. Effective organization, processing and presentation of the complex data required by these users is fundamental to realization of the main objective of Integer: efficient, accurate, configuration and monitoring of the entire technology stack from applications through the middleware and other layers on different types of servers, to the network infrastructure that supports the applications[2].

---

[1] The project, Integer, is an attempt to create a unified whole from the separate protocols, data elements and software systems we use to operate our increasingly complex computing environments. See: http://www.thefreedictionary.com/integer. Also see: http://en.wiktionary.org/wiki/integer#Latin

[2] In addition to information about the network elements collected from the systems themselves or algorithmically derived, the system will ultimately incorporate a good deal of metadata about the service elements in the system. For example, organizational information about the ownership or operations responsibility for components.

In addition, the rate of change of data in the system when deployed in even modestly sized environment can be very high. This scope of the dynamism[3], places additional demands not only on the server and persistence functions but also on the human interface if operators are to rapidly and effectively respond to developing conditions in advance of significant infrastructure or application failure.

These demands distinguish Integer from the more traditional eBusiness or administrative Web-based applications familiar to most people. One other dimension adds to the demands of the system in general and the user interface in particular, the range of user types. While the majority of the users are engineers of one type or another, there is great variation in type and depth of skill represented by these users. In addition, the system will be required to provide a number of simplified dashboards representing state and other information for less technical business owners of the systems under Integer's management.

Not a Technology Specification

No technologies are assumed in expressing the general requirements in this document, though some are referenced as a way of expressing different characteristics and how a requirement might be met. Often the use of one technology will not be consistent with others. The engineering process will have to resolve those issues during implementation.

Client Types and Environments

For the purpose of this discussion, the Integer user interface (client) software is intended to run on Mac OS X and Windows[4]. The server is constructed assuming a Linux environment[5]. Client diversity presents several challenges to the project. To the extent Web browsers are used, we will have to limit which versions we will support on the platforms we support.

Finally, operational personnel are increasingly using tablets and smart phones when 'on call'. It is expected that Integer will also have to provide at least some function in these environments as well. See the following section for limitations.

Not all Interfaces are Created Equal - Avoiding the Least Common Denominator

Support for the environments and client types described above does not mean equivalent function. In order for Integer to achieve the goals of efficiency, accuracy and timeliness described earlier, we will have to depart from the traditional least

---

[3] The dynamic nature becomes more pronounced when services are deployed in the cloud where infrastructure elements are created and removed algorithmically.

[4] As with all applications, the specific versions of the client platform will be limited due to engineering constrains. Specific versions will be detailed in release documentation. Depending on the method of implementation, an interface that runs on Mac OS X and Windows is also likely to run on most Linux systems.

[5] Specific versions will be enumerated in other documents.

common denominator function regarding user interfaces commonly employed in traditional eBusiness types of applications. This means that our core focus will be on optimization on one set of clients and the development of rich APIs that others[6] could use to optimize the user experience in other environments.

In an ideal world we would create a single native graphic user interface that is easy to use intuitive, powerful and visually appealing. To this writer, the environment that best exemplifies this is that of a native OS X application but that would require substantial refactoring to support other environments.

Desktops

The primary target environment for Integer functions is a 'traditional' desktop. That is, a device with sufficient screen real estate, graphics and CPU capacity to support the type of nearly instantaneous response expected when performing operations with direct manipulation.

Additionally, the larger display allows for presentation of more detail and context that an equivalent system with a smaller display. For the purpose of this discussion, a current generation Laptop with a 15" or larger display is treated as the same category of device as a desktop. Without limitation, every function available in Integer will be available via the desktop interface.

Tablets

Screen real estate and overall processing power are material differences that impact what we can reasonably expect from tablet implementations. Primary users of these systems will be on-call personnel and are less likely to need functions related to configuration and overall Integer operation they might need when not remote. For this reason, functions for tablets will initially focus on fault and performance display dashboards/interfaces. Users should be able to view in context configuration information and localized changes in specific configurations but they will not have the full complement of facilities for configuration changes and validation that the desktop interface provides. It is highly desirable that these systems have capability for reception of asynchronous messages from servers that may let operators know of faults or impending problems.

Differences between vendors makes it difficult to develop a cross platform application that maximizes the strengths of different platforms. While there are some technologies that ease the implementation across different tablet types such as Apple and Android based, they may force trade offs in performance, efficiency of implementation or other factors. The two first priorities for tablet activity (which are after the desktop interface) are to ensure APIs sufficient to support target functions

[6] One of the advantages of an open source effort is that affords the opportunity for interested parties to contribute software of value which would not otherwise be developed. Each environment may be a bit different so the functions and types of interface that matter most may vary. In Harvard's case, the primary desktop user for this application is assumed to be an OS X user, and the primary tablet is an iPad.

identified above and implementation of these functions on the Tablet platform most common in our environment.

Smartphones

The primary role of smartphones in the Integer ecosystem is that of status and notification receiver and event acknowledgment. Beyond the standard MMS and phone notification reception, a simplified event and status monitor based on the tablet versions will be supplied. In terms of priority, these devices fall after tablet support.

Users

As noted earlier, Integer must support a wide range of users, but one common element is that they are assumed to be generally technically capable people within their respective roles. This does not mean a clunky under-responsive interface is acceptable. To the contrary, users must have an interface that is powerful[7], responsive and flexible if Integer is to deliver the improvements in accuracy of configuration and overall efficiency gains that it promises.

Roles

The integration of functions across all dimensions of management (fault, configuration, accounting, performance and security) and the entire technology stack dictate that the system support many different users in a variety of roles. Examples include:

1. Network Engineer
2. Network Operator
3. Database Administrator
4. Application Engineer
5. Software Engineer
6. Application on-call engineer
7. Server/System engineer
8. Server/System operator
9. Tier 1-N support
10. Internal Consultants
11. External Consultants

In addition to these technically focused roles, some users will be in a management role and will require consolidation of information absent some of the detail more technical users require.

Flexibility

To serve the range of users described above, the system will have to offer considerable user interface flexibility. Based on role a number of options regarding

[7] See sections later in this document that discuss flexible, powerful, responsive, and intuitive user interfaces.

the graphic user interface should be provided (some are influenced by security/access control considerations) for example:

1. Options related to the types of action offered - for example configuration.
2. Scope of options performed - for example bulk reconfiguration may be permitted for some users in an organization but not available to others.
3. Views and types of data - this includes types of data such as performance data that may be overlaid on top of other data.

Power Users and a CLI

> *"If you want to teach people a new way of thinking, don't bother trying to teach them. Instead, give them a tool, the use of which will lead to new ways of thinking."*
>
> - Richard Buckminster Fuller

Yes, but if they won't eagerly use it, they won't get the benefit offered by the system, in this case, Integer. Operational personnel have, with good cause, over the years become skeptical about new software that claims to automate tasks and reduce errors. Systems and management interfaces that force an unwelcome interface or operational methodology have not been successful. For this reason, in addition to the graphic user interface described in this paper, a command line interface will be created that is a pass through to many systems for which that is the standard. The advantage is the Integer remains the central repository for information and these users can take advantage of the powerful graphic interface as they become more comfortable with system.

Interface Characteristics[8]

People often describe user interfaces as powerful, consistent, intuitive, easy to use, etc. The topics below describe the specific meaning/objectives for these terms in the context of Integer.

Powerful

A powerful interface is one that lets the user communicate to the system complex ideas or processes efficiently, often using direct manipulation[9]. The concept is that

---

[8] In addition to the specific references in the following sections, there are a number of articles of interest. Apple is widely regarded for their user interfaces. This article, by Kathryn Aragon in the Daily Egg discusses Apple guidelines for a better Web experience, but many points are relevant to all user interfaces.

[9] There are many on-line references and style guides available. The GNOME project has a very good site that discusses a number of important UI characteristics. This reference if for direct manipulation.

rather than having the user click through multiple screens[10] of data making selections (which can introduce error), the user interface, using techniques like direct manipulation and immediate feedback lets the user quickly and accurately accomplish their tasks, often on many systems at one time. Here is an example. One one portion of the screen are graphics that represent network elements and servers the support a mission critical application. On the left side of the screen is a palette of configuration policies with regard to access controls (ACLs) that permit certain protocols access selected systems from the Internet. In this case, the user only had to drag the policies from the palette to the selected systems on the right to cause them to be configured with the desired ACLs. This is a powerful user interface.

Adaptable to Different Skill Levels

Based on role, and the implied skill level of that role, the system should reconfigure the UI so that it is appropriate not only to the access permissions given to people with that role but also to the implied skill set. For example, certain people may not be permitted to have direct access to (re)configure core routers while other may in the event on an emergency.

Responsive[11] and Time Consistent

Responsive in this context means timely. What is timely depends on the context of the operation. For many years, three time values have described acceptable performance of user interfaces. The ACM published and article by Robert B. Miller in its proceedings: AFIPS '68 (Fall, part I) Proceeding of the December 9-11, 1968 fall joint computer conference, part I (Pages 267-277). It appears as if little has changed since then as these same numbers are commonly cited.

- If you want the user to have the feeling that they are directly interacting (that is responding immediately) with the system, about 0.1 seconds is about the limit.
- 1 second is the threshold for the users "flow of thought"[12] to remain intact.
- Anything longer than 10 seconds and the user begins to loose focus.

Time consistency refers to the length of time to perform the same/similar function in the system each time you use it. People can detect variations of 20%[13].

Consistent, Intuitive and Easy to Use

For the most part, these terms are straight-forward. The trick here is to not be a slave to consistency at the expense of intuitiveness. Sometimes doing something a bit different in a slightly different context can be helpful.

---

[10] Or using a command line interface on each of the managed systems to implement configuration changes.

[11] Google has done research as well in the context of Web searches, but it may also have applicability for this work.

[12] Response Times: The 3 Important Limits by Jakob Nielsen, January 1, 1993.

[13] Steve Sow, Ph.D., User Interface Timing Cheatsheet, Revision 0.2.0, Microsoft, Dec 12, 2006

Web Pages are the Wrong Paradigm

This heading may be heretical, but even though Web interfaces have improved over time and new technologies have promise, they do seem to generally cater to a least common denominator function. Even in those cases (and there are many) where Web based technologies have been applied to systems in the same domain as Integer, those interfaces have not tended to be as powerful, adaptable, responsive, time consistent, intuitive or easy to use as their native or Java thick client based alternatives.

It does not matter whether the difference is because of technology or ease/difficulty of implementing native client or Java thick client capabilities, the differences exist. The challenge for Integer will be to use the combination of technologies and approaches that represents the best compromise between an interface with the desired qualities and time/cost. This may be an area for experimentation and innovation before significant development on the user interface proceed.