Sapient
Computer Science 130: Team-Based Instant Messaging

September 26, 2001

CS 130: Team-Based Instant Messaging

About Sapient

Sapient, a leading business and technology consultancy, helps its clients discover and harness the competitive advantages that are possible in an increasingly digital, networked world. Founded in 1991, Sapient employs approximately 1,800 people in offices in Atlanta, Austin, Cambridge (Mass.), Chicago, Dallas, Denver, Düsseldorf, Houston, London, Los Angeles, Milan, Munich, New Delhi, New York, San Francisco, Tokyo, and Washington, D.C. Sapient is included in the Standard & Poor's (S&P) 500 Index.

Overview of Project

With the recent changes in the economic climate, Sapient has begun to implement new development methodologies and processes that allow for distributed development across multiple geographies. This new approach is advantageous to the company in many ways:

- Potential 24x5 development timeframes, shortening project lifecycles
- Significantly reduced development costs in specific regions
- Sharing of ideas across geographies builds each region’s knowledge & experience “bank”
- Wider base of potential client engagements, due to increased visibility within each geographic business arena

Of course, there are potential pitfalls that accompany cross-region development. Without a well-thought-out communication plan, and extensible mechanisms to support that plan, knowledge-share would be the least of our concerns. Synchronizing development efforts – even tracking the progress – of a complex, multi-layered and distributed project would become virtually impossible.

It is our hope that you, the contracted development team, will create an application tailored specifically to facilitate and encourage communication between our multi-region development teams. This application should be extensible and robust, but also simple and intuitive – designed for quick project team ramp-up on core functionality. Further, the application should be portable across platforms and leave a small-to-nonexistent footprint on each user’s development machine. We need an information-rich, elegantly presented, and real-time-enabled enterprise messaging system.

Business Objectives

As stated above, there are three main objectives that must be considered when creating the system. One, the system should be quick to learn. Second, the system should be easy to use. And lastly, the system should have a well-conceived interface.

Core Business Requirements

In order for the messaging system to be effective and provide value, the following features must be included in the system and available to all users:

Login and Identification

Each user of the system has a unique username and password that will be used to login into the “client” messaging system, and authentication will occur at the application server layer. The system must be able to validate each user before he or she is able to send or receive messages.

There should also be two different roles defined at the application server tier:

- User role one will be able to add, to view, to edit, and to delete personal information line items. This is the “administrator” role, and should have access to a separate application or data entry mechanism that allows for creating/editing/deleting data field definitions. See *Team Member Details* below for more information.
- User role two should be allowed to view information, transact messages, and update personal information exclusive to the currently logged-in user.

Team Member Details

Each team member will have a common “core” of information, regardless of project or user role. This information will be entered via either manual data entry, or, preferably, an automated import process.

Since Sapient already has an internal application that maintains certain key employee information, leveraging the existing data’s existence would vastly decrease the time required to initially install and configure the messaging application for a new (or re-structured) project team.

The core data is as follows:

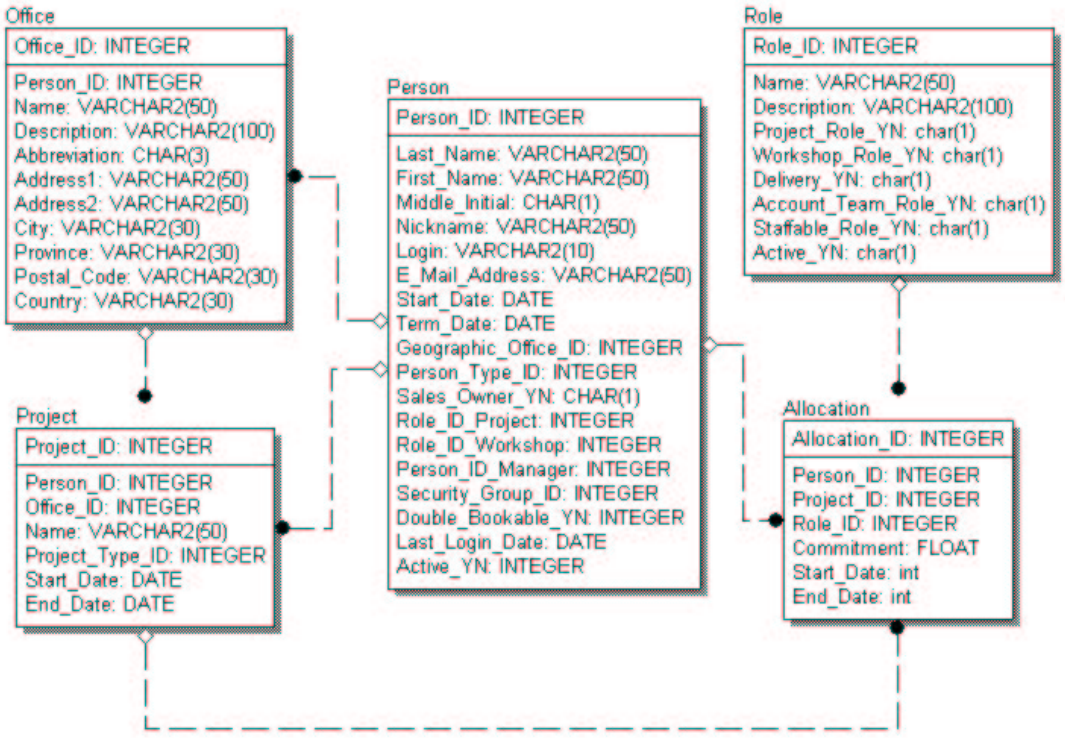
- Role – The role of the employee plays within the project (e.g. Engineer, Project Manager, Information Architect, etc).
- Name – The contact's full name.
- Start Date – The date the employee starts work on the project
- Stop Date – The date the employee ceases work (rolls-off) on the project
- Current Office – The physical office in which the employee currently works
- Email – The employee's business email address.
- Phone – The employee's current extension
- Project Name – The moniker given the current project
- Project Phase – The current phase of the project (e.g. Define, Design, Implementation, etc).
- Employment Status – FTE or Contractor (this impacts the user's accessibility to sensitive company information)

Additional information may be entered by the Project Team's Administrator to facilitate knowledge-share and eliminate confusion. This information should be flagged as either 'Required' or 'Optional'. Before a user can commence using the application, he or she MUST fill out all 'Required' fields with information. An example of 'Required' or optional information would be:

- Required: Sub-team name
- Optional: Birth-month and day
- Required: Technical/Functional expertise
- Optional: Current morale

This particular requirement suggests the need for a fairly generic and extensible data structure. This should be taken into account at the project design phase, but the development team will be given a fair amount of leeway when it comes to imposing upper bounds on the number of additional information fields allowed per project.

The below data model demonstrates the structure of existing data:



Optional Business Requirements

Although the above requirements lay out the *minimum* functionality required to implement and utilize a messaging system, there nonetheless exists a great breadth of functionality that would increase the usefulness, and therefore the appeal, of the application.

Instant Messaging

The highest priority of the “optional” feature list would be the application’s facilitation of real-time, PROACTIVE communication. This means that the messaging system’s implemented architecture should allow the user sending a message to elicit an immediate response from the “peer” application to which the message was directed. The *messaged* user should not be required to interact with the messaging interface prior to receiving the notification (apart, of course, from being logged-in and validated by the application).

Multi-User Chat

Employees may, for many reasons, need to interact with multiple individuals simultaneously. One reason for many-to-many interaction would be the scheduling of and online code review. This review would feature Developer A, seeking the constructive criticism of his code from

Developer B, C, and D. Allowing all parties to comment and view comments from other parties would significantly reduce the duplication of comments, and would also build knowledge among the developers – something that would be unlikely to happen if the review were to take place on three separate occasions, between three separate individuals.

Broadcast Messaging

Because there are commonly issues that arise over the course of a day that affect a great number of people, but don't necessarily warrant a team-wide email, the system should also allow for a 'broadcast' messaging functionality. The user should therefore be permitted to specify 1-to-many recipients of his or her message.

Queued Message Delivery

Because all users will not be online and accessible at all times, the logging and subsequent queued delivery of messages sent to an offline or invalid employee would greatly facilitate workflow management.

"External" Team Member Add

Although an employee would always be restricted from removing any existing 'core' team members from his or her active list, the ability to add and remove "external" team members from their messaging "buddies" could provide a great boon for team members allocated to a particular project, but scheduled to soon join another.

Simply expanding the breadth of communication and knowledge-share would invariably result in a better cross-cultural and cross-discipline exposure.

Encrypted Messaging Protocol

Should Sapient ever wish to migrate this application from an internal to an externally-visible communication mechanism, (that is, communication through the corporate firewall for the purposes of client participation or vendor checkups, etc), a secure messaging protocol would greatly enhance the extensibility of the application. Employees would not have to concern themselves with revealing sensitive client information to the world at large, and the client could take comfort in the knowledge that 156-bit algorithms protected his intellectual property from prying eyes.

Multiple Project Access

Many roles at Sapient are staffed to projects at a percentage less than 100%. A significant enhancement to the core functionality would be to give the employees who

overlap projects the ability to access project messaging systems where their allocation is greater than 0%. This would have to be built into the Security and Authentication module on the application server.

Technical Requirements

There are a few technical requirements for the system. The system should be a web-based application that we can add to our local Intranet and that any employee can use. Our Intranet is hosted on Microsoft Internet Information Server version 4.0 and built upon Active Server Pages. The data should be stored in a relational database system [ex. Microsoft Access]. Taking into account the requirements detailed above, we are concerned primarily with the functionality of the resulting application; less so with the tools and technologies used to produce it.

Usability Requirements
The application should be consistent with Sapient design standards. Please refer to our extranet, www.sapient.com, for a guide on how to design this application.

Project Logistics
The following people will be working with you to act as clients:

Shubo Mookerjee	smooke@sapient.com
Kimberly Tsuchida	ktsuch@sapient.com
Lori Holva	lholva@sapient.com
Bryce Calhoun	acalho@sapient.com
Nam Le	nle@sapient.com
Brock Meltzer	bmeltz@sapient.com
Philana Chen	pchen@sapient.com
Ron Chiu	rchiu@sapient.com

We will each take 1-2 teams and act as the main client to those teams. Once you are informed of whom your main client is, all questions should be emailed to that person. Although we will do our best to respond as quickly as possible, our responses might be delayed when we are overly busy at work (this only simulates the “real world” when clients are not available). Nevertheless, as I mentioned, we will do our best to respond promptly. Expect responses to come between 9 a.m. and 6 p.m. Also, it would be helpful if all questions emailed to us would come from the team’s main contact person. We look forward to working with all of you and wish you good luck in this class.