PTC Functional Specification

nSite Central

General Overview
Patterson nSite Central is a product suite allowing a CORPORATE office to connect, manage, view and report against any number of nSite Central or straight EagleSoft installations. One portion of this software package is web-based, allowing corporate USERS to access and view information from CLIENT locations, while the other portion is an agent-based data mining system responsible for the collection and retrieval of pertinent information.

Terminology
- **QUERY** – A QUERY is either a USER-supplied or automated request for data from a CLIENT or CLIENTs.
- **CLIENT** – A CLIENT is a group of computers located within the same dental office containing information solely relating to that office.
- **CORPORATE** – CORPORATE is a company that has purchased nSite Central in order to manage its CLIENTs.
- **USER** – A USER is any employee of CORPORATE who has an interest in viewing its financial records and/or client data.
- **ADMIN (ADMINISTRATOR)** – The ADMIN is an employee of CORPORATE who is responsible for system configuration, data-mining management, and overall system security, as well as creating and maintaining USER accounts.
- **ROOT** – ROOT is a hard-coded account built into every installation of nSite Central that allows Patterson’s Customer Support team to access any particular manifestation of the software to do disaster recovery and/or to view and manipulate data. ROOT is static and persistent across every installation of nSite Central.
- **REPORTING APPLICATION** – The REPORTING APPLICATION is a web-based graphical user interface that allows USERS to create QUERIES and view QUERY results.
- **CORPORATE I.S. MANAGEMENT APPLICATION** – The CORPORATE I.S. MANAGEMENT APPLICATION is a web-based graphical user interface that allows ADMIN to manage USERS, configure site settings (including security and data-mining settings), create QUERIES, and view QUERY results.
- **nSite Central** – nSite Central is the name used to refer to the overall software package detailed in this document.
- **EagleSoft** – EagleSoft is the name used to refer to the software package run at CLIENT locations.
- **CPDL (Centralized Performance Data Layer)** – The CPDL is a module that contains a database which stores the results of USER-submitted and automated QUERIES.
- **SECURITY PROFILE** – A SECURITY PROFILE is a group of preset access levels, which can then be assigned to one or more USERS.
**Functionality Targets**

This group of functional items is considered the minimum requirement specification for initial design and release.

- **nSite Central Registration** – The software package is installed by a Patterson Employee, after which the **ADMIN** account is configured. **ADMIN**’s password is hardcoded into the software, allowing for static login; this ensures that **ADMIN** is guaranteed login access under any/all circumstances.

- **CORPORATE I.S. MANAGEMENT APPLICATION Configuration (ADMIN Login)** – Using this interface, **ADMIN** can change all site settings.

- **USER Creation** – Any **USER** accounts must first be created by **ADMIN**. **USER**s may not create their own accounts. Upon creation, **USER** has no privileges; all subsequent privileges must be granted to **USER** by **ADMIN**.

- **USER Login** – Once a **USER**’s account has been created, the **USER** may login and use services for which s/he has access.

- **QUERIES** – Once a **USER** has **QUERY** privileges, the **USER** may submit a customized **QUERY** to the **CPDL**. The **CPDL** executes the **QUERY** and returns the results to the **USER** in table form.

**Additional Functionality**

This group of functional items is considered long-term goals but do not need to be present in initial design or release.

- **Scheduled QUERIES (Automated Reporting)**: A **USER** may subscribe to a **QUERY**, after which s/he will receive reports in PDF format automatically via email of the results of the **QUERY**.

**Excluded Functionality**

This group of functional items has been discussed thoroughly and is considered as not required.

- **Data Mining Agent System**: Data mining agents handle custom **QUERIES**, which are created by **USER**s to compile desired information from specified **CLIENT** locations.

**User Stories**

- **Create SECURITY PROFILE**
  1. **ADMIN** logs into **I.S. MANAGEMENT APPLICATION**.
  2. **ADMIN** navigates to security management page.
  3. **ADMIN** selects option to create new **SECURITY PROFILE**.
  4. **ADMIN** enters name of new profile in the form.
  5. **ADMIN** configures access levels for each securityItem.
  6. **ADMIN** saves changes.
  7. profileAccess table is updated appropriately.
  8. New **SECURITY PROFILE** is ready for use.

- **Security Item access level changed for SECURITY PROFILE**
  1. **ADMIN** logs into **I.S. MANAGEMENT APPLICATION**.
  2. **ADMIN** navigates to security management page.
3. **ADMIN** selects option to edit existing **SECURITY PROFILE**.
4. **ADMIN** chooses **SECURITY PROFILE** to edit.
5. **ADMIN** makes desired changes to access levels of securityItems he wishes to modify.
6. **ADMIN** saves changes.
7. **SECURITY PROFILE** is ready for use.

- **CORPORATE** adds new **ADMIN** account(s)
  1. **ADMIN** logs into **I.S. MANAGEMENT APPLICATION**.
  2. **ADMIN** navigates to **USER** management page.
  3. **ADMIN** selects option to create new **ADMIN**.
  4. **ADMIN** enters the following fields in a form:
     a. firstname
     b. lastname
     c. email
     d. password
     e. securityProfile (selected from a drop-down menu)
        o Administrator securityProfile selected for new **ADMIN**
     f. status
  5. **ADMIN** submits data to security database.
  6. New **ADMIN** is added to Employee table and employeeAccess table.

- **New USER account created**
  1. **ADMIN** logs into **I.S. MANAGEMENT APPLICATION**.
  2. **ADMIN** navigates to **USER** management page.
  3. **ADMIN** selects option to create new **USER**.
  4. **ADMIN** enters the following fields in a form:
     a. firstname
     b. lastname
     c. email
     d. password
     e. securityProfile (selected from a drop-down menu)
     f. status
  5. **ADMIN** submits data to security database.
  6. New **USER** is added to Employee table and employeeAccess table.

- **Check access levels via code**
  1. **USER** logs into **REPORTING APPLICATION**.
  2. System logic checks user's access levels for all security items.
     a. System logic checks "access" field for "securityItem" in the "employeeAccess" table for the **USER** ("employee" in the table).
     b. If access level is set to "true" for a particular securityItem, the user will have the privilege of using said securityItem and the **REPORTING APPLICATION** will display it for the user.
     c. If access level is set to "false" for a particular securityItem, the user will not have the privilege of using said
securityItem and the REPORTING APPLICATION will not display it for the USER.

3. REPORTING APPLICATION generates a dynamic page for the USER based on securityItem access levels.

- Perform USER authentication for login
  1. USER navigates to login page for system.
  2. USER enters "username" and "password", then clicks “Login”.
  3. System logic checks to see if USER is in security database.
     a. System logic checks to see if "userName" exists in "Employee" table.
        b. If username exists, system logic compares user-entered password with password stored in the "password" field for that particular userName.
           o If passwords match, system logic checks "status" field in "Employee" table.
              i. If status is true, user successfully logs into the system, and securityItem checking ensues.
              ii. If status is false, authentication fails.
           o If passwords don't match, authentication fails.
     c. If username doesn't exist, authentication fails.

- USER requests account from ADMIN
  1. CORPORATE sees need to add an additional USER account.
  2. New USER submits desired username and password to CORPORATE, which gets forwarded to ADMIN.
     a. If username already exists, ADMIN will select a new username for USER that has not been previously used.
     b. ADMIN will retain USER's original password.
  3. ADMIN logs on to CORPORATE I.S. MANAGEMENT APPLICATION.
  4. ADMIN navigates to USER management interface.
  5. ADMIN selects option to create new USER.
  6. ADMIN enters the following fields in a form:
     a. firstname
     b. lastname
     c. email
     d. password
     e. securityProfile (selected from a drop-down menu)
     f. status
  7. ADMIN submits data to security database.
  8. New USER is added to Employee table and employeeAccess table.
  9. New USER account is ready for use.

- USER subscribes to reports
  1. USER wishes to receive reports of the results of automated QUERIES via email.
  2. USER logs on to REPORTING APPLICATION.
  3. USER navigates to automated QUERY section.
  4. USER selects QUERY or QUERIES s/he is interested in following.
5. **USER** submits request through a form.
6. **REPORTING APPLICATION** returns a page with either a "Subscription successful" or a "Subscription failed" message.
7. **REPORTING APPLICATION** sends **USER** a confirmation email if successful.

- **ADMIN** deletes **USER** account
  1. **USER**’s account is no longer valid for any number of reasons:
     a. **USER** has done something worthy of account deletion.
     b. **USER** was fired from **CORPORATE**.
     c. **USER** is no longer an employee of **CORPORATE**.
     d. **USER**’s account has been compromised.
     e. Any other reason deemed appropriate by **ADMIN**.
  2. **ADMIN** logs on to **CORPORATE I.S. MANAGEMENT APPLICATION**.
  3. **ADMIN** navigates to page that displays a complete list of **USERS** for **CORPORATE**.
  4. **ADMIN** selects account of **USER** that was deemed worthy of deletion.
  5. **ADMIN** deletes **USER**’s account.
  6. **USER**’s account is no longer present within **nSite Central**.

- **nSite Central** data mining service fails, **ADMIN** still logs in
  1. One of several possible failures occurs:
     a. The data mining management system of **nSite Central** fails.
     b. **REPORTING APPLICATION** fails.
  2. **ADMIN** logs on to **CORPORATE I.S. MANAGEMENT APPLICATION**.
  3. **ADMIN** views error logs to determine the problem.
  4. **ADMIN** determines the appropriate course of action.

- **ADMIN** creates connection with **CLIENT**
  1. **CORPORATE** wishes to add a new **CLIENT** to **nSite Central**.
  2. **ADMIN** logs on to **CORPORATE I.S. MANAGEMENT APPLICATION**.
  3. **ADMIN** navigates to connections page, which displays list of all **CLIENTs** currently connected.
  4. **ADMIN** clicks "Add new client" button.
  5. **ADMIN** is taken to a new page with a form to be completed.
  6. **ADMIN** supplies necessary information.
  7. **ADMIN** submits form.
  8. New **CLIENT** connection is added to overall **CLIENT** connection list with its current connection status.

- **ADMIN** changes **USER** access levels
  1. One of several possible reasons exists for **USER**’s privileges to be changed:
     a. **USER** desires his/her settings to be changed.
     b. **CORPORATE** deems it necessary to change privileges for a particular **USER**.
  2. If **CORPORATE** approves privilege modifications for a **USER**, permission is given to **ADMIN** to make the changes.
  3. **ADMIN** logs on to **CORPORATE I.S. MANAGEMENT APPLICATION**.
  4. **ADMIN** navigates to **USER** management page.
5. **ADMIN** selects option to view/edit **USER** access levels.
6. **ADMIN** chooses **USER** for which to change access levels.
7. **ADMIN** does one of two things:
   a. He selects a default security profile from a drop-down menu (profiles taken from "profileAccess" table) that will apply preset access levels to all security items for that particular employee.
   b. He manually goes through each security item for that particular employee, modifying the access levels as he sees fit (the "access" field in the "employeeAccess" table will be modified).
8. **ADMIN** saves the changes.
9. Security access levels for that particular **USER** are now changed.

- **USER** login
  1. **USER** wishes to access **REPORTING APPLICATION**.
  2. **USER** navigates to login page of **REPORTING APPLICATION**.
  3. **USER** enters username and password.
  4. **USER** is now logged in to **REPORTING APPLICATION**.

- **USER** creation of QUERIES
  1. **USER** wishes to send a **QUERY** to **CPDL**.
  2. **USER** requests appropriate privileges from **ADMIN**.
  3. Upon receiving necessary privileges from **ADMIN**, **USER** logs in to **REPORTING APPLICATION**.
  4. **USER** navigates to **QUERY** creation interface.
  5. **USER** creates a custom **QUERY** and submits it to **CPDL**.
  6. **CPDL** returns the results to **USER** in table form.
Modules Affected By Scope

- N/A:

Report Additions

- QUERY Reports: QUERY Reports are reports generated by USER-supplied QUERIES that are displayed in table form on a Web page within the REPORTING APPLICATION.
  a. Filtering / Retrieval Needs
     o Filter by CLIENT location
     o Filter by Date Submitted
  b. Report Categories
     o N/A

- Automated Reports: Automated reports are scheduled reports generated from the results of automated QUERIES performed by data-mining agents. USERS may subscribe to automated reports and receive them by email in PDF form.
  1. Description
     a. Filtering / Retrieval Needs
        o N/A
     b. Report Categories
        o N/A

Report Modifications

- N/A:
User Security Additions

- **Submit QUERY**: This level of security ensures that any `QUERY` submitted to the CPDL is properly formatted and will not produce any malicious results.
  1. Check Access:
     a. **USER** List: Check `QUERY` privileges for **USER** who is submitting the `QUERY`.
  2. Dependant Security:
     a. N/A
  3. Additional Product Listing:
     a. **EagleSoft**
  4. License Needed:
     a. `nSite Central, EagleSoft`
  5. Default Value:
     a. Disallow

Departments / Representatives

- **Development**
  1. Jim Garrett, Software Development & Quality Assurance Manager
Approval & Sign Off

The undersigned members of the PTC Development Team agree upon this design and accept it in its entirety as a functional target.

Software Development & Quality Assurance Manager  

Product Manager  

DBA / Data Architect  

Senior Quality Assurance Technician  

Senior Software Engineer  

Senior Software Engineer  

Senior Software Engineer (optional)  

Senior Software Engineer (optional)
# Review / Revision History

<table>
<thead>
<tr>
<th>Revision Number</th>
<th>Changes Made / Reviewed</th>
<th>Effective / Reviewed Date</th>
<th>Revision Author</th>
</tr>
</thead>
</table>
| 1.00            | Initial Revision                                                                        | 10/21/08                  | Michael Dunn
|                 |                                                                                         |                           | Ryan Sessions
|                 |                                                                                         |                           | Kyle Kerrigan             |
| 2.00            | Several new user stories added, some changed. New definitions added. Minor grammatical  | 11/17/08                  | Michael Dunn
|                 | changes made throughout document.                                                        |                           | Ryan Sessions
|                 |                                                                                         |                           | Kyle Kerrigan             |