Chemotherapy Order Management System (COMS)

Technical Manual

Version 2.2

September 2015

Department of Veterans Affairs

Revision History

Date	Revision	Description	Author
9/4/2015	2.2	Finalized Prototype Version	Sean Cassidy
3/5/2015	2.1	Updated Content	Sean Cassidy
2/20/2015	2.0	Initial Prototype Version	Sean Cassidy
9/27/2012	1.2	Proof of Concept Acceptance Version	Sean Cassidy
9/15/2012	1.1	Revised Edition	Sean Cassidy
4/6/2012	1.0	Initial Version	Sean Cassidy

Table of Contents

List of Figures

1. Background and Introduction

The Veterans Health Administration (VHA) has one of the largest cancer populations in the country; it is also the fastest growing group of VHA patients. VHA provides oncology services at more than 100 different locations by integrated oncology care teams consisting of, but not limited to, a physician/provider, nurse, and pharmacist. Teams typically provide care on an outpatient basis, although some patients may require hospitalization and inpatient services.

A uniquely high-risk and high-complexity domain of health care, Oncology services support has not been effectively implemented within the existing VHA Electronic Health Record primarily due to the lack of functionality required for the specialty. VHA's oncology processes are a mix of paper-based and computer-based practices, presenting potential error, adverse events, and inefficiencies. This creates a clinical environment with minimal standardization and limited direct order entry of chemotherapy. For these reasons, the VHA Office of Health Information (OHI) Patient Safety Workgroup rated this issue as having a high level of patient safety risk. Accordingly, an initiative within VHA's Innovations Program sought to enhance the clinical environment and safety for oncology patients through development of the Chemotherapy Order Management System (COMS) application as part of VHA's Strategic Incubation.

The COMS application enhances the clinical environment and safety for oncology patients through development and implementation of an automated ordering and management process available within VHA's clinical practice setting. In either an outpatient or inpatient setting, the COMS application supports the unique needs of oncology healthcare teams with standardized capabilities to meet direct order entry, clinical documentation, and assessing the administration of chemotherapy. COMS provides interoperability with VHA's electronic health record, interfacing and interacting with existing applicable systems, modules, capabilities, and processes within Computerized Patient Record System (CPRS) graphical user interface and Veterans Health Information Systems and Technology Architecture (VistA) databases.

The web-based COMS application consists of an interface via Hypertext Precursor (PHP), Java Script, Node.js, Simple Object Access Protocol (SOAP), and Representational state transfer (REST) web services. The application provides five clinical modules – Chemotherapy Template Order Source, Order Entry Management, Treatment Documentation, Flow Sheet, and End of Treatment Summary – and miscellaneous functionality that collectively serve to deliver robust functionality to support users in executing their roles and responsibilities in various oncology care processes. Within current legacy system functionality, the COMS application provides provider order entry and promotes patient safety via read/write interoperability with the electronic health record. The application uniquely offers exportability of chemotherapy templates for national vetting and proliferation to facilitate VHA-wide standardization of chemotherapy regimens for oncology services and patients across the VA enterprise.

2. Orientation

This COMS Technical Manual is for use in conjunction with VHA's COMS application. It outlines technical information for the COMS application to enable use at VHA facilities. The

intended audience of this manual is Office of Information and Technology (OI&T) staff responsible for the proper technical support of VHA clinical applications.
This manual provides an overall explanation of COMS from a deployment, installation, and production operations perspective with the assumption that the reader is familiar with the following:

- CPRS/VistA computing environment
- Internet and database server environments
- Microsoft operating environment

This manual does not detail installation or user operation aspects of the COMS application. Such topics are available in the COMS Installation Guide and COMS User Manual, respectively.

3. Application Overview

COMS automates chemotherapy ordering and documentation while accommodating local facility policies for clinical preferences/processes and implements several VA defined system-wide protocols. These protocols include a national set of chemotherapy order templates, standardization of the calculation method for medication dosage and dose rounding, and a standard documentation format for chemotherapy treatment plans, administration, assessments, and summaries. COMS creates and manages chemotherapy templates; clears and places medication orders in VistA; enables treatment documentation; displays a temporal flow sheet for relevant clinical data, medications administered, and user assessment of response to treatment; and creates a treatment summary for the current healthcare team, referring/primary care providers, and other clinical and support staff. The COMS application fulfills legal/professional requirements, fosters Joint Commission compliance and enhances patient safety with VistA/CPRS interoperability and documentation ultimately stored in VA's electronic health record.

This manual instructs users on the technical aspects of COMS application functionality, navigation of the application, and effective use of COMS to record and reflect treatment provided to oncology patients. COMS consists of five clinical modules: Chemotherapy Template Order Source (CTOS), Order Entry Management (OEM), Treatment Documentation (TD), Flow Sheet (FS), and End of Treatment Summary (EoTS) plus miscellaneous functionality.

3.1. Chemotherapy Template Order Source (CTOS) Module

The CTOS module permits the oncology provider to download a chemotherapy regimen template from a central library and modify it for local use. It also enables authorized users to create original, new templates or create a template from an existing one, while maintaining version control. User-created templates are available immediately for local use and ultimately for national use upon review and inclusion in the central library.

Each template provides pre-therapy, therapy, and post-therapy information including recommended medications; dosages and parameters for dosing; total number of cycles; number

of days and administration days within each cycle; medical references; medication reminders for the regimen; and information on the number of patients currently undergoing treatment with the template (for existing templates). CTOS incorporates standardized template naming conventions with version control, acceptable medical terminology and abbreviations, and other applicable medical guidelines to reflect the existing care practices. At any time, users may review clinical practice guidelines and references relevant to the template supporting the provision of chemotherapy for a specified treatment regimen.

The CTOS module affords flexibility to enable the user to assign a unique, user-friendly template name. These templates may be applied to a particular patient record to generate an order sheet for the regimen and used throughout the remaining modules of the COMS application. During the process of applying a template to a patient, the oncology provider has the opportunity to identify the effective date; select the body weight and body surface area formula to use for medication dosage calculations; categorize the regimen for the patient as curative or palliative care; indicate whether the patient/regimen are part of a clinical trial and specify the name of the clinical trial, if applicable; identify patient amputation(s); and document the current performance status of the patient. Although users may only apply one template to a patient at any given time, all templates previously applied to the patient remain available for review within the Treatment Regimens and Summaries panel.

3.2. Order Entry Management (OEM) Module

The OEM module permits the oncology provider to prescribe, modify, or print an order for pre-therapy, therapy, and post-therapy medications from any template currently applied to a specific patient. When a provider applies a template via the CTOS module, COMS generates an order sheet and sets in motion the individualized chemotherapy regimen for curative or palliative care of oncology patients. Oncology providers utilize the OEM module to tailor templates – currently applied to patients based on diagnosis and other considerations – as a prescription for an appropriate course of therapy to meet chemotherapy treatment goals.

While CTOS templates provide a general regimen for a specific diagnosis and establish the direction for treatment, the OEM module enables oncology providers to customize the medications within each template and order sheet for each individual patient. Primary instances of customization include modification to medications, dosages, or administration dates. The OEM module supports the oncology provider in tailoring applied template medications on any particular date of administration. For any future administration date within the prescribed regimen, oncology providers may edit, cancel, or hold medications after the initial order and change the administration date for a specific date or future dates in the treatment cycle or regimen.

The OEM module enables oncology providers to record the current performance status of the patient to reflect improvement/decline throughout the regimen. At any time, users may view medication dosages for orders and any associated body surface area (BSA)-based calculations for dosing. Rarely changed during a treatment regimen, body weight (e.g. actual, ideal) and selection of body surface area (BSA) calculation methodology (e.g. Dubois, Mosteller) may be changed through the Patient Information panel or by applying a new template through the CTOS module. Since many templates use specified body weight and BSA methodology to calculate

chemotherapy medication dosages, the capability to make changes further enables oncology providers to convert generalized templates into individualized treatment plans for each specific patient throughout the regimen cycle.

The OEM module facilitates communication among the healthcare team as the medication orders progress from "ordered" through "administered" status.

- After the CTOS module initiates an "ordered" status, the order may be cleared in accordance with local facility policies/practices and then viewed by the pharmacist.
- Pharmacy modifications to the order – considering local facility rounding rules for medication dosage calculation and identifying the rationale – are communicated to the provider or identified for provider re-signature, as local procedures warrant.
- Once "finalized" between the provider and pharmacist, orders are transmitted to the Veterans Health Information Systems and Technology Architecture (VistA) instance for pharmacy preparation. Cancelled or held orders are not transmitted to VistA, but remain within OEM for visibility of the initial order and subsequent action.
- Following pharmacy preparation and dispensing of the medication, a nurse may administer it to the patient, effectively recording the order as "administered" in the Treatment Documentation (TD) module.

This OEM workflow – complemented by COMS messaging functionality – facilitates and documents dialogue among the oncology provider, pharmacy, and nursing staff as it relates to modifying the patient's chemotherapy regimen order and any subsequently required provider approval for changes to the original order. Authorized users may print the order at any time.

Altogether, the OEM module enables oncology providers to tailor pre-therapy, therapy, and post-therapy medications for each individual patient and facilitates treatment plan communication and coordination among the healthcare team.

3.3. Treatment Documentation (TD) Module

The TD module permits the oncology nurse to view relevant historic and current clinical data and document the provision of chemotherapy and nursing assessments of the patient since the previous administration and throughout the current treatment. These assessments of adverse reactions or no adverse reactions may be entered into COMS – regardless of administration day or day of rest – for inclusion in the Flow Sheet (FS) and consideration for End of Treatment Summary (EoTS).

Following the Chemotherapy Template Order Source (CTOS) establishment of the regimen and Order Entry Management (OEM) customizing medications for each individual patient, the TD module facilitates the crucial role of documenting the administration of the regimen's prescribed medications as finalized and dispensed. Pre-populated data from the OEM module serves as the foundation for recording medications administered (Note, held or cancelled medications remain visible, but unavailable for administration documentation). The attending nurse documents doses administered to the patient, including dates, exact minute start/stop times, administration comments, patient and dose verification, symptom assessments, and infusion reactions.

The TD module supports the oncology nurse with a chemotherapy/biotherapy header and six activity-specific panels to convey information previously obtained and to facilitate the documentation of new information relevant to the administration of prescribed medications.

- General Information – Provides laboratory results and historic vital signs in standard and metric values; enables the nurse to document patient verification (from two sources) and confirmation consent documentation is on file, patient teaching, dual verification of medication dosing, and vital signs obtained that day.
- Assessment – Enables the nurse to document the top chemotherapy symptoms and clinical grading of those side effects (using Common Toxicity Criteria terminology, as loaded in Site Configuration) plus any other symptom the patient has encountered since the previous administration, identify those to trigger an adverse event alert, or record no adverse reactions.
- IV Site – Facilitates nurse documentation of the intravenous (IV) site access date; device, needle gauge, and delivery mechanism; body location and site appearance; as well as verification of the patient's brisk blood return before, during, and after treatment.
- Administration – Enables the nurse to annotate the administration of pre-therapy, therapy, and post-therapy medications, as sequenced in the applied template regimen. These medications are pre-populated from the OEM module once orders are cleared for preparation and medications are dispensed by the pharmacy. In support of clinical practices and guidelines, "positive action" by the nurse is required to properly document medication administration (held or cancelled medications are identified, but not available for administration documentation).
- Infusion Reactions – Facilitates documentation of details for three common chemotherapy infusion reaction categories – Extravasation, Cytokine-Release Syndrome, and Hypersensitivity/ Anaphylaxis – and any other reaction to the administration of medications on the treatment day.
- Discharge Instructions – Facilitates communication of specific clinic information; key regimen, medication, or side effects information; and patient reminders for the next administration day, and scheduled laboratory tests. Consistent with CPRS documentation templates, enables the nurse to record patient education topics that include teaching methodology/preferences, attendees, and other key aspects of discharge instructions.

Altogether, the TD module provides users robust functionality to document individual assessment, treatment, and instructions provided for oncology patients.

3.4. Flow Sheet (FS) Module

The FS module offers a snapshot of care for the healthcare team to view relevant clinical data; the disease response/toxicities/side effects and other general narratives for patient reaction to the chemotherapy; an overview of administered/held/cancelled medications; and pertinent laboratory results. Through direct entry of information and display of specific information from OEM and TD modules, the FS module provides the healthcare team with an efficient, user-tailored display of relevant information and patient-centered documentation of chemotherapy administration.

The FS module supports direct entry of disease response, toxicity/side effects, – using Common Toxicity Criteria terminology, as loaded in Site Configuration – and other general information annotations. Authorized users may enter detailed, free text comments regarding the tumor response to the administered medications from the regimen prescribed in the CTOS module; customized, held, or cancelled in OEM module; and administered as documented in the TD module. Fields dedicated to toxicity/side effect selection, grade, details, and free text entry during a particular administration day or rest day within the regimen provide insight into patient reaction to the chemotherapy agents. Users may also enter an uncategorized "other" comments to clearly communicate observations and recommendations. Members of the healthcare team may view disease response, toxicity/side effects, and other comments through dedicated panels within the FS module at any time throughout the regimen.

The COMS application retrieves and displays relevant laboratory results from the electronic health record (VistA and CPRS). In this manner, FS supports rapid view of laboratory results within the context of the prescribed regimen, administered/held/cancelled medications, and other significant clinical documentation. The FS module also automatically retrieves and presents pre-therapy, therapy, and post-therapy medication administration specifics from the TD module or held/cancelled actions processed in the OEM module. With administration day columns, the FS module presents relevant information for medications and dosages prescribed and subsequently held, cancelled, or administered to the specific patient. Eliminating dual entry, COMS directly populates this documentation from the OEM module or the Administration panel within the TD module, respectively. Members of the healthcare team may view this overview as a snapshot of held/cancelled/administered medications for each administration day throughout the regimen, selected cycles, or specified date(s).

The first section of the FS module enables user entry while the remaining sections concisely display patient-centric, regimen-specific information for the healthcare team.

- General – Provides date, patient performance status, and weight; enables entry and viewing of disease response, toxicity/side effects, and other comments relevant to the patient's treatment.
- Pre-Therapy – Presents specific pre-therapy medication administration details for each administration day; automatically populates from the OEM module for held/cancelled medications and the TD module's Administration panel for administered medications.
- Therapy – Displays specific administration details of prescribed chemotherapy agents for each administration day; automatically populates from the OEM module for

held/cancelled medications and the TD module's Administration panel for administered medications.

- Post-Therapy – Presents specific post-therapy medication administration details for each administration day; automatically populates from the OEM module for held/cancelled medications and the TD module's Administration panel for administered medications.
- Disease Response Panel – Provides all disease response narratives in reverse chronological order, as entered through the "Add General Information" functionality.
- Toxicity/Side Effects Panel – Displays all toxicity/side effects information – toxicity title, grade, details, and narrative – in reverse chronological order, as entered through the "Add General Information " functionality or TD module's Assessment panel or Infusion Reactions panel.
- Other Panel – Presents all other narratives in reverse chronological order, as entered through the "Add General Information " functionality
- Laboratory Results – Displays laboratory results relevant to the provision of oncology services; automatically populates from CPRS.

Altogether, the FS module provides an efficient display of relevant information and patient-centered documentation of chemotherapy administration.

3.5. End of Treatment Summary (EoTS) Module

The capstone module of the COMS prototype is the End of Treatment Summary (EoTS). The EoTS module supports the oncology provider to stop a treatment regimen and create the summary of care rendered, including results achieved throughout the specified treatment regimen. The EoTS module also enables the healthcare team to view completed treatment summaries. Following the conclusion or discontinuation of a regimen and its applied template, the provider typically generates the treatment summary through EoTS functionality. To aid in generation of the summary report, the EoTS module retrieves relevant information from various COMS modules and patient-specific panels then pre-populates several sections of the treatment summary for provider consideration when preparing narratives for provider report sections.

The EoTS module supports pre-population of patient and regimen details, type of cancer, vital signs data (in standard and metric values), body surface area information, clinical trial, allergy, performance status, and medications administered. Further retrieval from the Flow Sheet (FS) module enables the provider to review healthcare team entries throughout the regimen regarding disease response, toxicity side effects, and other general information for creation of the summation narrative and categorization of the disease response. The EoTS module also supports an overall provider report with free text entry to communicate patient and regimen specific assessment to the current and future healthcare team. This information is contained within COMS and transmitted to the patient's electronic health record for healthcare team members without COMS access.

The EoTS module treatment summary worksheet provides five main sections to guide the provider through generation of the End of Treatment Summary.

- Pre-populated Components – Provides pre-populated data for patient and regimen details, type of cancer(s), amputation(s), initial and final vital signs information for the regimen, body surface area factors and values, clinical trial, allergy, performance status, and medications administered to the patient throughout the regimen.
- Patient Disease Response – Enables the provider to review FS module entries to categorize the disease response (complete response, partial response, minor response, progression, or stable) and create summation narrative.
- Toxicity Side Effects – Enables the provider to review FS module and Treatment Documentation (TD) module assessments for Toxicity Side Effects and create summation narrative.
- Provider Report – Supports free text narrative for provider to enter patient and regimen specific overall assessment as a summary for the chemotherapy treatment, results, and/or prognosis.
- Follow-Up Appointments – Supports free text narrative for provider to enter information relevant to the patient's follow-up appointments.

Altogether, the EoTS module facilitates stopping a treatment regimen and provides a chronological history of diagnosis, treatment, changes in treatment, disease response, and patient outcomes.

3.6. Miscellaneous Functionality

COMS miscellaneous functionality is the underpinning that supports the five clinical modules and general application performance. These clinical modules and general application capabilities are influenced by COMS miscellaneous functionality not specifically associated with any particular module, but integral and supportive of overall operational effectiveness. Miscellaneous functionality is classified as non-administrative – patient specific functionality and general functionality – and administrative.

Patient specific functionality includes patient relevant information and history where users may view patient/regimen details and view or enter (with intelligent data detection) the patient's vital signs. General functionality for orders within the application and messaging affect all members of the healthcare team utilizing COMS from order creation to medication administration. Extensive administrative functionality exists for COMS administrators to setup and maintain the application's database contents, accessibility, and interoperability while tailoring COMS support to accommodate local facility policies for administrative and clinical preferences and processes.

COMS Miscellaneous Functionality is provided through 11 non-module sections.

- Patient Information Panel – Expandable/collapsible display of patient-/regimen-specific information, including patient demographics, body surface area details, template, regimen status, type(s) of cancer, allergies, and clinical trial retrieved from various COMS modules and Veterans Health Information Systems and Technology Architecture (VistA) legacy application

- Medication Reminders Panel – Expandable/collapsible display of regimen-specific reminders for adjunct therapy or other considerations relevant to the selected patient to include title, description, and prescribed timing throughout the regimen
- Adverse Events History Panel – Expandable/collapsible display of patient-specific information regarding toxicities and/or adverse events documented throughout oncologic treatment to include the event, grade, details, comments, date of observation, and designation for flagged as an alert, as appropriate
- Patient Vitals Panel – Expandable/collapsible display of chronologic vital signs data (in both standard and metric values, as appropriate) retrieved from various COMS modules and VistA; permits documentation of patient vital signs – within intelligent data entry parameters, if established – with or without a template applied
- Laboratory Information Panel – Expandable/collapsible display of chronologic laboratory test results retrieved from VistA for viewing within COMS
- Orders Tab – Location for all active orders within the application that facilitates and monitors progression of medication orders through the continuum of ordered, cleared/in-coordination, finalized (or held or cancelled), dispensed, and administered (note, administered and in-coordination require precursor actions and are not available for direct selection); displays chemotherapy orders for administrations over the next three calendar days for all patients within the COMS instance
- Messages Tab – Communication hub for application messaging associated with provider notifications for information, alerts, or action
- Template List Tab – Location for all treatment regimens/templates within the application for viewing and printing plus current total of patients undergoing treatment with each regimen
- Template Promotion Tab – Restricted functionality for Local Template Manager (LTM) users and Central Template Authoring Group (CTAG) to promote templates from individual user to local to national template availability
- Reports Tab – Location for users to generate and view inventory, patterns of care determination, and laboratory reports
- Site Configuration Tab – Robust core for administrative functionality to support database setup, maintenance, and interoperability; manages capabilities for documentation lists and contents (clinic information, discharge instructions, lookups, medication documentation, and toxicity); template management (delete template, disease staging, emetic medications, neutropenia/emesis risks, and import/export templates); user access (lockout and user roles); clinical decision support (cumulative dose medications and intelligent data entry); and facility preferences (active workflows, IV fluid types, medication holds, medications not rounded, rounding rules, pharmacy management, and signature verifications).

Altogether, miscellaneous functionality enables the application to support standardized direct order entry, healthcare team coordination, documentation of chemotherapy treatment, and clinical module and overall application capabilities.

4. Architecture

The COMS application is comprised of two separate but interconnected components. The first is the backend or database component while the other, frontend component serves as the user interface. The database was developed in PHP and is responsible for all interaction with the Structured Query Language (SQL) Database, the second component is Node.js and it's use of existing Remote Procedure Call (RPC) calls through the RPC Broker to execute routines within VistA for the saving/retrieval of patient records, chemotherapy templates, medications, and pharmaceutical orders. The user interface was developed in JavaScript and provides all the controls, windows, and forms for user interaction to create, edit, and view the information necessary for COMS to function effectively and efficiently.

4.1. Model-View-Controller (MVC) Framework

Both the user interface and database components were developed using the Model-View-Controller (MVC) pattern. This is a software design pattern used to separate the presentation of information from the underlying architecture and business rules.

The "Model" portion of the MVC pattern defines the data and business rules. The "Controller" portion manages the input and converts it to instructions for either the "Model" or "View" portion. The "View" portion is the output representation or display of the data in terms the application user can understand, such as panels and tabs of information.

4.2. Ext JS

Ext JS is a JavaScript library and application framework developed by Sencha (http://www.sencha.com/products/extjs/) for use in designing robust user interface centric, web-based applications. For the COMS application, the Ext JS library permits division of the application into the MVC pattern with the panels, forms and tabs for the end user, defining various data models for communicating with a backend component and the controls necessary for the application to communicate with the backend database.

4.2.1. General Note

Any textual value available in a Drop Down control (also termed a "Combo Box" or simply "Combo"), for example the name of a medication, will be stored as the Globally Unique Identifier (GUID) for the record containing that textual value from the appropriate table. In this manner, if a textual value needs to change (e.g. due to a typographical error in the string), none of the saved records pointing to that textual value would need to change.

4.2.2. Ext JS MVC Software Design Pattern

- A **model** is an object representing data or business rule. Models persist through the data package and may link to other models through associations.
- A **view** serves as a window displaying the data. Grids, trees, and panels are all views.
- A **controller** serves as the "glue" between the model and the view, controlling the timing and passing of models to views for application display. A controller contains all the

programming code for rendering views, instantiating Models, and any other application logic.

- **Event handlers s**hould be included as part of a Controller for the timing and passing of models. Views do not contain event handlers; they only fire the events.
- **Hierarchy of controllers** determines the handling of information. Each controller contains dedicated views and models. When designing interaction between controllers, developers consider application wide events. For example, if the View in Controller1fired an event to be handled by Controller2, the application is designed to for Controller1 to fire a generic application event and Controller2 to handle it for updating its views. In this instance of interaction between the two controllers, Controller1 has a higher/earlier hierarchy followed by Controller2.
- **Validations** are contained within controller models to ensure the data is consistent with designed expectations. If the data is consistent, the model validation will pass and permit the controller to View the information for display. If the data is not consistent with designed expectations, the controller's model should invalidate the data and not provide a View for the user.
- **Model specific data manipulations** are similar to validations and are contained within the Controller's Model. Accordingly, the application manipulates the data according to designed specifications before the Controller permits a View for the user.
- **Data querying logic** must also reside within a model. For a model as an abstraction to the system that processes input routed by controllers and updates its state, it serves as a proxy class for the backend database system. All queries to the database should go through a model. Any business processes in the application should be formulated within a model only.

4.2.3. Standard Model Template

A standard model template is used when there are no dependencies for the model. Within COMS, an example of a standard model template is as follows:

```
Ext.define("COMS.model.<Name of Model>", {
        extend: "Ext.data.Model",
        alias: "model.<Name of Model>",
        fields: [
                "Field1",          // default field type is string, no name/type object required
                "Field2",
                { "name" : "Field3", "type" : boolean },
                ...
        ],
        "proxy": {
                "type": "rest",
                        // The following parameters should be set to false, to ensure that no
                        // additional parameters are passed to the service call
                "filterParam" : false,
```

```
"groupParam" : false,
"pageParam" : false,
"startParam" :false,
"limitParam" :false,
        // The following parameter prevents the “noCache” parameter
        // from being passed to the service call, however, the service call code
        // itself should always ensure that it sends current data back.
        // Set HTTP Headers to NoCache???
"noCache" : false,
        // Note, there is an additional parameter which might get passed
        // to the service call in the case of a model being used by a combo box.
        // This is the “query” parameter which must be set to false in the
        // ComboBox config itself, e.g. “queryParam” : false
api: {
        read: <READ URL>,               // uses HTTP GET
        update: <UPDATE URL>,           // uses HTTP PUT
        create: <CREATE URL>            // uses HTTP POST
},
"reader": {
        "type": "json",
        "root": "records",
        "successProperty": "success",
        "totalProperty": "total",
        "messageProperty": "message"
},

// Any functions which are specific to the proper retrieval/handling/validating of
// the data should be included in the model file itself.
“afterRequest” : function (request, success)  {
        if ("read" === request.action) {
                this.readCallback(request);
        }
        else if ("create" === request.action) {
                this.createCallback(request);
        }
        else if ("update" === request.action) {
                this.updateCallback(request);
        }
        else if ("destroy" === request.action) {
                this.deleteCallback(request);
        }
},

“readCallback” : function (request)  {
        if (!request.operation.success) {
                Ext.Msg.show({
                        title: "Warning",
                        msg: "Could not load Data. Please try again.",
```

```
                                        buttons: Ext.Msg.OK,
                                        icon: Ext.Msg.WARNING
                                });
                        }
                }
        }
});
```

4.2.4. Dependent Model Template

Conversely, a dependent model template is used when a given model contains an array of objects defined in a separate model. The following is an example of a dependent model template used within the COMS application:

```
Ext.define("COMS.model.<Name of Model>", {
        extend: "Ext.data.Model",
        alias: "model.<Name of Model>",
                // If this model is dependent on one or more additional models
                // then include the "uses" array
        uses: [
                "COMS.model.<Dependent Model Name>"
        ],

        fields: [
                "Field1",       // default field type is string, no name/type object required
                "Field2",       // This is an array of <"Dependent Model Name>" objects
                { "name" : "Field3", "type" : boolean },
                ...
        ],
                // The "hasMany" property defines which field(s) is/are dependent
                // on which additional models
        hasMany: {
                model: "COMS.model.<Dependent Model Name>",
                name: "Field2"
        }
        "proxy": {
                "type": "rest",
                api: {
                        read: <READ URL>,               // uses HTTP GET
                        update: <UPDATE URL>,           // uses HTTP PUT
                        create: <CREATE URL>            // uses HTTP POST
                },
                "reader": {
                        "type": "json",
                        "root": "records",
                        "successProperty": "success",
                        "totalProperty": "total",
```

```
                "messageProperty": "message"
            }
        }
});
```

4.2.5. Source Model Template

A source model template is used when a given model is used by a dependent model for an array of objects defined in a separate model. Model specific functions should be included in the Model file. Within the COMS application, an example of a source model template is as follows:

```
Ext.define ("COMS.model.<Name of Model>", {
        extend: "Ext.data.Model",
        alias: "model.<Name of Model>",
            // If this model is related to one or more additional models
            // then include the "uses" array
        uses: [
            "COMS.model.<Name of Model which is dependent>"
        ],
        fields: [
            "Field1"
        ],

        // The " belongsTo" property defines which model(s) are dependent upon this one
        belongsTo: {
            model: "COMS.model.< Name of Model which is dependent >"
        }
        "proxy": {
            "type": "rest",
            api: {
                read: <READ URL>,           // uses HTTP GET
                update: <UPDATE URL>,       // uses HTTP PUT
                create: <CREATE URL>        // uses HTTP POST
            },
            "reader": {
                "type": "json",
                "root": "records",
                "successProperty": "success",
                "totalProperty": "total",
                "messageProperty": "message"
            }
        }
});
```

4.2.6. Standard Store Template

The COMS application also utilizes a standard store template. This section presents the standard store template technical aspects and utility.

```
Ext.define('COMS.store. <Name of Store>", {
        extend : 'Ext.data.Store',
        model : <Name of Model>
});
```

Note: The "autoLoad" option should not be used. Additional options are declared after the model.

Simplified method to access a data store by name

The COMS application accesses a data store by name through its database "get" functionality in the backend component.

If a store is included in the list of stores for a controller:

stores : ["Store1", "Store2"])

Then simply reference the store by name via the controller's "get" function:

this.get<StoreName>Store();

For example:

this.getStore1Store();
this.getStore2Store();

However, this will only work if the store is declared in the stores list. For example given the above stores list one can call ***this.getStore1Store(), this.getStore2Store()*** but a user may not call ***this.getStore3Store()*** until declared in the stores list.

4.2.7. Standard View Template

The COMS application utilizes a standard view template in the frontend/user interface component.

```
Ext.define("COMS.view.<Application Section Folder Name>.<View Name>" , {
        "extend" : "Ext.tab.Panel",
        "alias" : "widget.<View Name>",
        "name" : "<View Name>",
        "autoEl" : { tag : "nav" },
        "items" : [
                { "xtype" : "AddLookups", "title" : "Add LookUps" },
                { "xtype" : "container", "html" : <HTML Content> }
        ]
});
```

Views are designed and accomplished through individual JavaScript (JS) files in a folder descriptive of the application section where they are located (e.g. Patient Tab, Orders Tab, and Template Authoring Tab). Each view will contain a unique Alias and Name and will utilize Hypertext Markup Language (HTML) 5 tags (e.g. section, header, footer, navigation) to promote better document structure. A complete list of standard view templates is available at http://www.w3schools.com/html/html5_new_elements.asp .

4.2.8. Standard Controller Template

The COMS application also uses standard controller templates. Similar to the views, controllers are designed and achieved through individual JS files in a folder descriptive of the application section where they located in (e.g. Patient Tab, Orders Tab, and Template Authoring Tab). An example of standard controller templates within the application is as follows:

```
Ext.define("COMS.controller. <Application Section Folder Name>.<View Name>" , {
"extend" : "Ext.app.Controller",
        // Include any stores, views and Data Models managed by this controller
"stores" : [ "LookupStore" ],
"views" : [ "Management.AdminTab" ],
"models" : ["LookupTable"],

        // Include any local references needed by this controller
        // a local reference can be accessed via the "get" function of the controller
        // e.g. this.getLookup()
"refs" : [
        { "ref" : "<Ref Name>",  "selector" : "<Selection Query>" },
],

"init" : function() {
        // Assign any event handlers to specific objects and events
        this.control({
                "<Selection Query>" : {
                        select : this.<Local Function Name>
                },
        });
},

        // define the functions used in this controller to manage the specific events.
  "<Local Function Name>" : function(<Parameters>){
  }
})
```

4.3. Web Services

The World Wide Web Consortium (W3C, located at http://www.w3.org/) defines a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network". COMS utilizes two distinct type of Web Services; Simple Object Access Protocol (SOAP) and REpresentational State Transfer (RESTful).

The COMS application uses RESTful Web Services to transfer information between the frontend/user interface and the backend/database components. A RESTful Web Service allows a simple structure to exchange information between these two components. For the user interface to obtain information from the database component, the web browser sends an http "GET"

request to the database component via a simple Uniform Resource Identifier (URI) or "web address" such as http://coms.va.gov/Patients/viewall

This web address communicates with the database component to retrieve a list of all patients in the system and to return specific information about those patients in the form of a JavaScript Object Notation (JSON) record, as follows:


```
{
   "success": true,
   "total": 41,
   "records": [
      {
         "id": "B521F525-6099-E111-8812-000C2935B86F",
         "name": "PATIENT  FOURHUNDREDFIFTYFIVE",
         "DOB": "04/07/1935",
         "Gender": "M",
         "Age": "77",
         "DFN": "100455",
         "TemplateName": "2012-3-0001-ABCD-PACLITAXEL INJ,CONC   200-20120711",
         "TemplateDescription": "NSCLC - Paclitaxel Single Agent",
         "TemplateID": "BFF16C4E-74CB-E111-A078-000C2935B86F",
         "TreatmentStart": "07/11/2012",
         "TreatmentEnd": "10/03/2012",
         "Goal": "Curative",
         "ClinicalTrial": "",
         "WeightFormula": "Adjusted Weight",
         "BSAFormula": "Haycock",
         "PAT_ID": "AA5760E7-79CB-E111-A078-000C2935B86F",
         "BSA_Method": "Haycock",
         "PerformanceStatus": "1",
         "TreatmentStatus": "On-Going - Rest Day",
         "Amputations": []
      },
… 40 more records similar to the one above
   ]
}
```


The above JSON object communicates the model portion (specifically the "COMS.model.PatientInfo" model) of the user interface that the call to retrieve the patient information was successful, that it returned 41 patient records and then lists each of the 41 individual patient information records. The actual retrieval of the patient information from the database component is the task of the PHP code. The PHP code uses its own models, views and controllers to query the SQL Database and/or VistA database via Node.js to obtain the information and (using the model's data specific data manipulator) combine it into the data sent to the user interface.

Actions to save a new record to the database component are done via an http "POST" request. This action is also achieved via a simple URI with the data inside a JSON object. COMS accomplishes requests to update an existing record via an http "PUT" request.

COMS uses the following service calls to communicate between the user interface and database components:

Service Name: ActiveWorkflows - Retrieve a list of the currently active workflows
Service Name: AddCTOS - Apply a Chemotherapy Template to a specified patient
Service Name: AddEoTS - Create a new End of Treatment Summary Record
Service Name: AddFlowSheetRecords - Update the changes made to the current Flowsheet to the patient's record for this treatment regimen
Service Name: AddLookup - Add new Data to the Lookup Table
Service Name: AddND_Assessment - Add a new Nursing Documentation Assessment record to the patient's record for this treatment regimen
Service Name: AddND_GenInfo - Add a new Nursing Documentation General Information record to the patient's record for this treatment regimen
Service Name: AddND_IVSite - Add a new Nursing Documentation IV Site Assessment record to the patient's record for this treatment regimen
Service Name: AddND_React_Assess - Add a new Nursing Documentation Reaction Assessment record to the patient's record for this treatment regimen
Service Name: AddND_Treatment - Add a new Nursing Documentation Treatment Assessment record to the patient's record for this treatment regimen
Service Name: AddPatientTemplate - Apply a new Template to a patient
Service Name: AddVitals - Add a new Vitals record to the patient's record for this treatment regimen
Service Name: AdminGlobals - Return a list of global variables
Service Name: AdminUsers - Return (or create a new) list of Users allowed to access the system
Service Name: Allergies - Return a list of all Allergies for a specified patient
Service Name: CTCAE_Data - Return the Common Terminology Criteria for Adverse Events data; retrieved from http://evs.nci.nih.gov/ftp1/CTCAE/CTCAE_4.03_2010-06-14.xls
Service Name: CTCAE_SOC - Return the Common Terminology Criteria for Adverse Events (CTCAE) System Organ Class (SOC) groupings as listed in the National Cancer Institute Center for Bioinformatics documentation
Service Name: CTOS - Return the Template Data for the specified Template
Service Name: DeleteLookup - Delete specific data from the Lookup Table
Service Name: DeleteTemplate - Delete a specific Treatment Template from the application
Service Name: DiseaseStage - Return list of all Stages used in "Select Disease Stage" drop down menu fields
Service Name: DiseaseType - Retrieve a list of disease types, used in the "Select Disease Type" drop down menu fields
Service Name: DrugRegimen - Return the information for a particular drug regimen (e.g. drug, dosage, and route of administration)
Service Name: DrugUnits - Return a list of drug units for use in the Units drop down menu fields
Service Name: Drugs - Return a list of drugs for use in the Drugs drop down menu fields
Service Name: Edit_OEMRecord - Edit a specific Order Entry Management (OEM) record to change a specific treatment order
Service Name: EmotegenicLevel - Return a list of Emotegenic Levels for use in the Emotegenic Level drop down menu fields

Service Name: EoTS - Return a specific End of Treatment Summary
Service Name: FlowSheetRecords - Return a specific Flow Sheet for a specific patient
Service Name: FluidType - Return a list of Fluid Types for use in the Fluid Types drop down menu fields
Service Name: HydrationDrug - Return a list of medications for use in the medication drop down menu fields
Service Name: Infusion - Return a list of Infusion Types for use in the infusion type drop down menu fields
Service Name: LabInfoResults - Return a list of all laboratory results for a specified patient
Service Name: Lookups - Generic service to return records for a specific LookupType in the Lookup Table

Possible Lookup Types are included in the following table:

Lookup Type ID	Lookup Type	Description
1	Diagnosis	Diagnosis Type Lookup
2	Drug	Drug Type Lookup
3	TreatmentIndicator	Treatment Indicator Selector Values
4	Regimen	Template Selector Values with Template Name in Description
5	TIProtocol	Treatment Indicator Protocol Selector Values
6	DiseaseCat	Disease Category Selector values
7	DiseaseType	Disease Types (aka Cancer Types) with Abbreviations in Description
8	DCBlood	Disease Blood Category Selector values
9	References	LookUp for References with URI in Description
10	PerformanceStatus	Patient's Performance Status with Sequence in Description
11	Unit	Medication Unit Measurement
12	Route	Regimen Route Type
13	Emetogenic	Emetogenic Level
14	LabTest	Laboratory Test Types
15	Health Care Provider	Health Care Provider Type
16	Specimen	Specimen Type
17	Lab Test Site	Laboratory Test Site
18	TimeFrameUnit	Various Time Frame Units
19	Total_Courses_Max	Lists the maximum number of treatment cycles that may be repeated for a given regimen
20	Cycle_Length_Max	Lists the maximum length (days/weeks) of any cycle within a treatment regimen
21	MasterTemplateRefXRef	Cross Reference from Master Template to References Lookups
22	TemplateSource	Location of Templates
23	DiseaseStage	Disease Stage

24	User	User Name/ID
25	TemplateAlias	Alias for template name
26	NonFormaDrug	Add Non-Formulary Drug
27	PatientAllergies	Allergies For a Patient
28	FluidType	Fluid Types
29	Allergies	Type of Allergy
30	PatientAmputations	Amputations for a Patient

Service Name: NodeMatch - Return patient information from the VistA via Node.js and RPCs for the patient specified by the Data File Name (DFN) ID

Service Name: Patient - Start a search in the VistA to retrieve all the latest information on the specified patient to store in the COMS database

Service Name: MedsNonRounded - Return a list of medications which should NOT be rounded when calculating medication dosages

Service Name: ND_Assessment - Return a specific Nursing Documentation Assessment record from the patient's COMS record for this treatment regimen

Service Name: ND_GenInfo - Return a specific Nursing Documentation General Information record from the patient's COMS record for this treatment regimen

Service Name: ND_IVSite - Return a specific Nursing Documentation IV Site Assessment record from the patient's COMS record for this treatment regimen

Service Name: ND_React_Assess - Return a specific Nursing Documentation Reaction Assessment record from the patient's COMS record for this treatment regimen

Service Name: ND_Treatment - Return a specific Nursing Documentation Treatment Assessment record from the patient's COMS record for this treatment regimen

Service Name: OEMRecords - Return list of Order Entry Management (OEM) Records for specified patient

Service Name: Orders - Return a list of all current pharmaceutical orders in the COMS application

Service Name: PatientTemplate - Return a list of Template information for treatment regimens previously applied to the specified patient

Service Name: Patients - Retrieve information for a list of or specific patient used in the "Select Patient from CRPS" drop down menu fields

Service Name: ReadND_Treatment - Returns the Nursing Documentation Treatment records from the patient's COMS record for this treatment regimen

Service Name: Reasons - Retrieve a list of reasons for a workflow

Service Name: References - Retrieve a list of references for template definitions used in the authoring of a template

Service Name: SavePatient - Save all the current information for the current patient; does not include personally identifiable information (PII)

Service Name: Templates - Return list of templates available for specified source (i.e. National, Local, or My Templates)

Service Name: TimeFrameUnit - Return a list of Time Frame Units (days, weeks, months) which are used in the "Cycle Length" drop down menu fields

Service Name: Vitals - Return list of Vital Signs (e.g. blood pressure, height, and weight) for specified patient

COMS uses RESTful based Web Services to communicate between the user interface and database components. However, the Node.js system utilizes SOAP based Web Services to present its information to any requesting service, such as COMS.

4.4. Node.JS and VistA Remote Procedure Calls

COMS uses a set of RESTful services calls that uses Node.js to bridge a data connection between COMS and VistA via the RPC Broker. The Node.js service calls are a suite of web services that exposes medical domain functionality, Medical Domain Objects (MDO). Node.js provides a web service function to allow an issuing VistA Remote Procedure Call (RPC).

These web services provide the client application developer with common SOAP web services and documentation necessary to create client applications. In turn, this incorporates the organization's business rules and provides access to data from multiple VistA instances and other disparate data sources. Node.js fosters COMS interoperability with VistA/CPRS and serves as a critical link in overall COMS functionality within the VHA clinical environment.

4.5. VA FileMan

COMS utilize SSH2's ability from within PHP to connect to a VistA instance and transmit the generated data for the patient and pharmacy orders. After the connection channel is established, COMS executes VA FileMan commands to navigate through VistA and execute commands to perform standard "silent" routines. These "silent" routines define, enter, and retrieve information from the associated VistA instance for use within the COMS application and posting of information into the VistA database's electronic health record.

4.6. Databases

The COMS application uses MS SQL Server for storage of local data. COMS sends data and retrieves data from the SQL Databases via PHP based RESTFul Web Service Calls. All data calls into the SQL Server are made via Stored Procedures designed in accordance with industry standard leading practices to reduce the probability of SQL Injection attacks and data corruption.

4.7. Service Calls – General Information

The default URI for all service calls is specified entirely in lower case as shown in the following example:

Valid:
http://example.com/patients

Invalid:
http://example.com/Patients

Service calls return data as a valid JSON object as specified in Request For Comment (RFC) #4627 (http://www.ietf.org/rfc/rfc4627.txt), as shown in the following example:

Valid:
{ "name": "Simon" }

Invalid:
{ name: "Simon" }
{ 'name': "Simon" }
{ "name": 'Simon' }

Additionally, service calls issued with an http “GET” request have two basic return conditions. The return condition may be either, a “successful” return or a “failure” return as follows:

Service call returning a “successful” return condition:

{
"success" : true,
"total" : INT - representing the # of records returned
"records" : [] - array of 1 or more records
}

Service call returning a “failure” return condition:
{
"success" : false,
"message" : "" - message detailing the reason for the failure
}

If the service call was unable to return any records because none matched the search criteria, then a message of "No Records Found" should be returned with a success of false.

As such, all stores should be configured with a “REST” proxy and a “JSON” reader:

```
   "proxy": {
      "type": "rest",
      "reader": {
         "type": "json",
         "root": "records",
         "successProperty": "success",
         "totalProperty": "total",
         "messageProperty": "message"
      }
   }
```

Note: When data is POSTed back to the server via any JSON Object, the service will automatically include the time/date stamp of the posting. The service will also include the User ID of the user who initiated the data POST back to the server.

This information is not listed in any Ext JS models as it is generated by the backend service call.

4.8. Application Breakdown

4.8.1. Patient Tab

The Patient Tab is the default view when accessing the COMS application. It is used to access patient-specific functionality, but not required to access Orders, Template Authoring, Template List, Messages, or Site Configuration functionality (Note, availability of various tabs is dependent upon user role).

4.8.2. Patient Selection

The COMS application provides three different methodologies for patient selection. Users may accomplish patient selection by Administration Date(s), CPRS Query, or Patient Selection from CPRS, as shown in **Figure 1**.

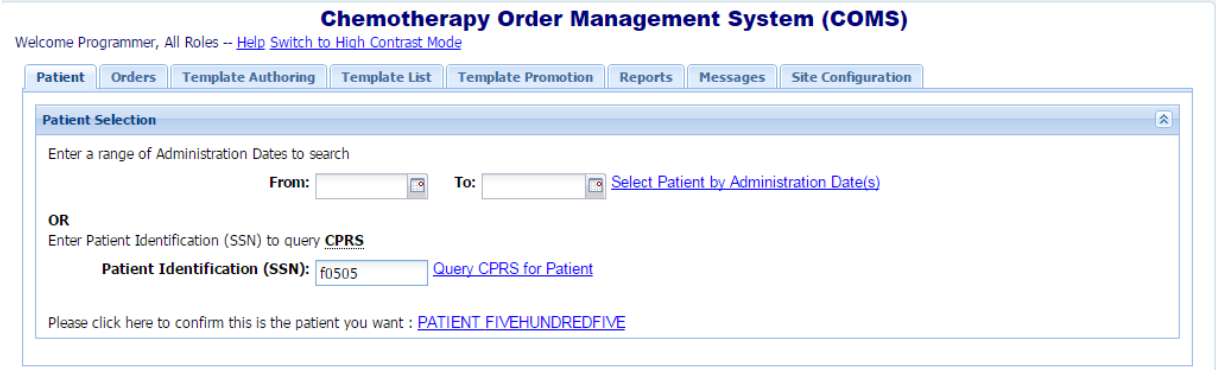

Figure 1: Patient Selection

Select Patient by Administration Date(s)

To select a patient by administration date(s), COMS utilizes the "Patients" Service Call. The application passes one or more dates (the "From" date if only one date passed) and will return a list of patients who have Administration Dates within the specified date range. By default, the service will return all patients who have future Administration Dates.

Options for the Patients Service Call include the following:

- http://example.com/patients – Return all patients who have Administration Dates ≥ today
- http://example.com/patients/mm_dd_yyyy – Return all patients who have Administration Dates ≥ the date passed
- http://example.com/patients/mm_dd_yyyy/mm_dd_yyyy – Return all patients who have Administration Dates ≥ the first date passed but ≤ the second date passed

Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientList" Model.

```
PatientList = [              // Array of Patients used to select a specific patient.
        {
                "PatName" : "", // Patient Name
```

```
                "PatID" : "",       // GUID to identify this specific patient
        }
]
```

Query CPRS for Patient

The "CPRSPatient" Service Call is used to query the VistA instance using the CPRS Data File Name (DFN). This provides a Patient Index for Node.js to return a single Patient Record in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientList" Model. If the DFN passed does not uniquely define a single patient, the service call will return a list of patients for selection. The CPRS Patient Service Call for the COMS application is as follows:

- http://example.com/cprs_patient/##### – Return a specific patient whose DFN matches that passed. If the DFN passed is not complete, CPRS will return all patients matching the partial DFN.

Select Patient from CPRS

If either the "Patients" service call or the "CPRSPatient" service call returns a single patient, COMS presents a link to confirm the specified patient, as shown in **Figure 2**. Otherwise, COMS presents a selection or ComboBox to select a specific patient from the returned list.

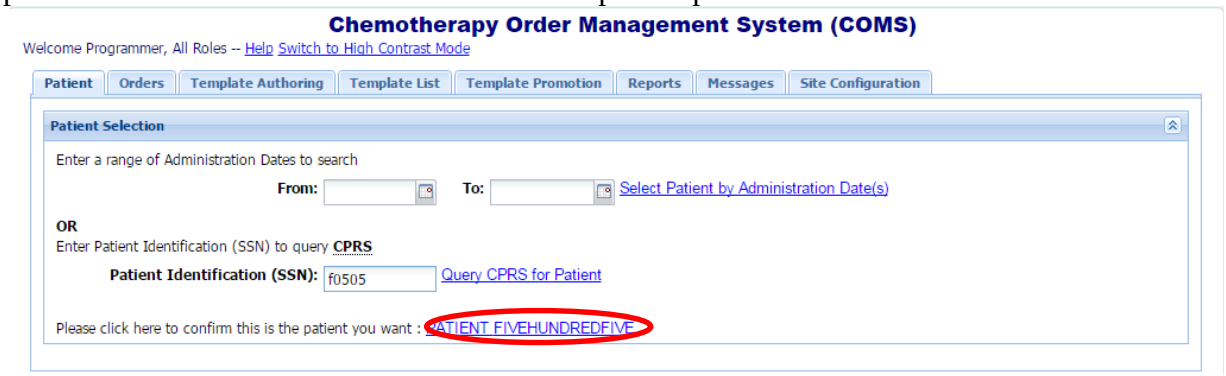

Figure 2: Confirmation of Returned Patient

Regardless, selecting/confirming a specific patient will launch a series of service calls to retrieve all COMS and VistA information for the specified patient.

- http://example.com/patient/data/GUID - Retrieve basic details for specified patient
- http://example.com/patient/vpr/GUID - Retrieve all current VistA Patient Record data for specified patient
- http://example.com/patient/labs/GUID - Retrieve laboratory results for specified patient
- http://example.com/patient/vitals/GUID - Retrieve vital signs history for specified patient
- http://example.com/patient/template/GUID - Retrieve template history for specified patient
- http://example.com/patient/oem/GUID - Retrieve Order Entry Management Results for specified patient
- http://example.com/patient/template_data/GUID - Retrieve details on the current template applied to the specified patient

The data returned by this series of service calls is used to populate the individual sections of the Patient Tab as depicted in the following sections.

4.8.3. Patient Information

The "Patient Information" section of the "Patient" tab contains multiple expandable/collapsible panels to display data returned from the series of executed service calls. **Figure 3** depicts the information within expanded panels displayed from the returned service calls.

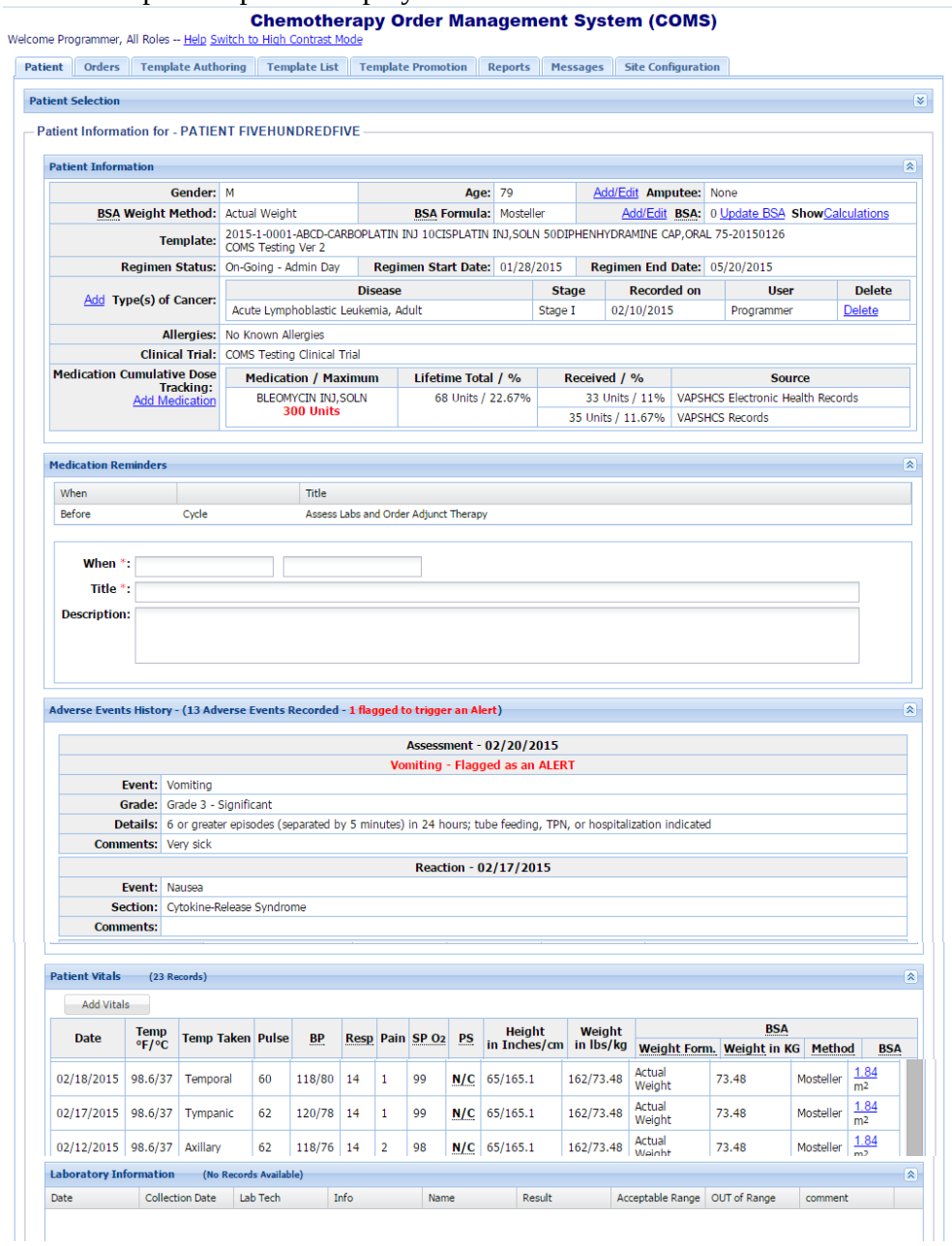

Figure 3: Patient Information from Service Calls

Data Service Call

The "NodeJSData" service call issues a series of internal service calls to the Node.js system to retrieve any VistA/CPRS data for the specified patient. The "NodeJSData" service call only returns the status of the success/failure of the service calls placed to Node.js to the frontend. All data retrieved is stored in the COMS internal database on the backend. An example of the Node.js data service call is as follows:

- http://example.com/patient/NodeJS/GUID

The GUID passed to the "NodeJSData" service call is the "PatID" (consisting of the first letter of the patient's last name and last four digits of the patient's social security number). This is obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientInformation" Model. The "NodeJSData" service call returns a standard message body as follows:

```
{
        "success" : true,          // could return false if the service call failed
        "message" : " NodeJS Msg: <Details on the status of the web service calls made>"
}
```

Patient Data Service Call

The "Patient Data" service call returns some basic information from VistA/CPRS about the selected patient for use in the COMS application. The patient data service call is composed as follows:

- http://example.com/patient/data/GUID

The GUID passed to the "Patient Data" service call is the "PatID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientInformation" Model, as follows.

```
PatientInformation = {          // This is a single element data structure
        "PatName" : "",         // Patient Name
        "PatID" : "",           // GUID to identify the specific patient
        "PatTreatID" : "",      // Patient Treatment ID - uniquely identify all information
                                // related to this particular treatment regimen
        "Gender" :  "",         // Patient Gender (M/F)
        "Age" : "",             // Patient Age (in years) - integer
        "Amputations" : [       // Any amputations the patient has
                {
                        "Amputation" : "",      // Amputation Description –
                                                // derived from the allowable amputations table
                        "PctBodyMass" : ""      // % of body mass the amputation removed,
                                                // used in BSA Calculations
                }
        ],
```

```
        "BSA_WeightMethod" : "",        // BSA Weight Method –
                                        // derived from the allowable BSA weight methods table
                                        // specified when template applied
        "BSA_Method" : "",              // BSA Method - derived from the allowable BSA
                                        // methods table specified when template applied
        "BSA_Value" : "",               // BSA Value, calculated based on available criteria
        "BSA_Calculations" : "",        // Actual formula used to derive the current BSA Value
        "PatHeight" : "",               // Patient's Height (in Inches) used for the BSA Calculation
        "PatWeight" : "",               // Patient's Weight (in Pounds) used for the BSA Calculation
        "TemplateID" : "",              // Template ID – GUID to identify the specific template
                                        // used in this treatment regimen
        "TemplateDesc" : "",            // Template description
        "TemplateName" : "",            // Template name
        "TemplateAppliedDate" : ""      // Date the template was applied
        “OEM_ID” : “”                   // GUID for the OEM Record for this patient/treatment
        "RegimenStatus" : "",           // Regimen Status for the current calendar day –
                                        // derived from the Regimen Status table
                                        // (On-Going - Admin Day, On-Going - Treatment Day,
                                        // Started, Ended, ???)
        "NextAdminDate" : "",           //Future administration date derived from template
        "RegimenStartDate" : "",        // Date the Regimen was started (First day of the first cycle,
                                        // Not necessarily the date the template was applied)
        "RegimenEndDate" : "",          // Date the Regimen was originally calculated to end
                                        // (based on cycle duration and number of cycles)
        "RegimenStopDate" : "",         // Date the Regimen was stopped (if it was stopped
                                        // prematurely)
                                        // this would be different than the Regimen_EndDate)
        "CancerTypes" : [               // Types of cancer the patient has
                {
                        "Type" : "",    // Cancer Type - derived from list of cancer types
                        "Stage" : ""    // Stage - derived from the list of stages for each
                                        // particular cancer type
                }
        ],
        "Allergies" : [                 // Any allergies the patient has,
                                        // if the patient has NO allergies then this field is ""
                {
                        "Name" : "",    // Name of the allergy
                        "Type" : "",    // Type of allergy (food, medication, etc)
                        "Comment" : "" // Any comments in the patients record
                }
        ],
        "ClinicalTrial" : ""        // If this is a clinical trial, then this would be the
                                    // description of the trial, or it is ""
}
```

Within COMS, the "Patient Information" data is rendered in the "Patient Information" Panel via an Ext JS library xTemplate. After rendering the Patient Information, links may be available for

Body Surface Area (BSA) calculations and the currently applied template. If a template has been applied to that patient, links are available for "Show Calculations" and "Open Template". The "Show Calculations" link will display a pop-up with the calculations performed to calculate the BSA. The "Open Template" link will display the currently applied Template information in the "Chemotherapy Template Order Source" tab.

Patient Treatment History Service Call

The "Patient Treatment History" service call returns a list of previously applied templates with links to treatment history/results associated with the applied templates. If there is a template currently applied to the patient this template information is also returned as the "CurrentTemplate". The patient treatment history service call is as follows:

- http://example.com/patient/treatments/GUID

The GUID passed to the "Patient Treatment History" service call is the "PatID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientTreatmentHistory" Model, as follows.

```
PatientTreatmentHistory = [      // This is an array of individual Patient Treatment History JSON Objects
        {
                "CurrentTemplate" : True/False,
                "PatTreatID" : "",                  // Patient Treatment ID
                                                    // GUID to uniquely identify all information related
                                                    // to this particular treatment regimen
                "TemplateID" : "",                  // Template ID - GUID to identify the specific
                                                    // template used in this treatment regimen
                "TemplateDesc" : "",                // Template description
                "TemplateName" : "",                // Template name
                "TemplateAppliedDate" : ""          // Date the template was applied
                "RegimenStatus" : "",               // Regimen Status for the current calendar day
                                                    // derived from the Regimen Status table
                                                    // (On-Going - Admin Day, On-Going - Treatment Day,
                                                    // Started, Ended, ???)
                "RegimenStartDate" : "",            // Date the Regimen was started (First day of the
                                                    // first cycle, Not Necessarily the date the template
                                                    // was applied)
                "RegimenEndDate" : "",              // Date the Regimen was originally calculated to
                                                    // end (based on cycle duration and number of cycles)
                "RegimenStopDate" : ""              // Date the Regimen was stopped (if it was stopped
                                                    // prematurely this would be different than the
                                                    // Regimen_EndDate)
                                                    // If "Regimen_StopDate" = "" then the treatment is
                                                    // still going on or treatment has been stopped
                "EoTS_ID" : ""                      // GUID for the End of Treatment Summary Record,
                                                    // if "EoTS_ID" = "" then EoTS has not been generated
        }
]
```

The "PatientTreatmentHistory" data is rendered in the "Treatment Regimens & Summaries" panel via an Ext JS library xTemplate.

After rendering the Treatment Regimens & Summaries, each template will have the following links displayed:

- Show Details
- Stop Treatment
- Generate End of Treatment Summary (only for historical templates)
- Show End of Treatment Summary (only for historical templates with completed summary)

Show Details: Selection will display the details of the Template and treatment provided information in the following COMS clinical modules within the Treatment Details panel:

Chemotherapy Template Order Source – Template Details
Order Entry Management – Individual orders for treatment
Treatment Documentation – Details of the treatment performed and patient assessment
Flowsheet – High level overview of treatment performed and snapshot of ongoing care

Stop Treatment: Available for the Current Template only, selection will prompt the user to confirm that the treatment the patient is currently undergoing is to be stopped. If the user confirms stoppage, COMS will flag the treatment as stopped, and this link will be removed.

Generate End of Treatment Summary: Available when a treatment has ended but an End of Treatment Summary (EoTS) has not been generated, selection of this link will initiate the EoTS and guide the user through the process of generating an End of Treatment Summary. The treatment summary will be captured in SQL and written to VistA as a progress note via Node.js and RPC calls.

Show End of Treatment Summary: Available when an End of Treatment Summary has been generated, selection of this link will display a read-only version of the End of Treatment Summary in a popup window. The treatment summary is also available within the VistA patient record as a progress note.

Patient Vitals Service Call

The "Patient Vitals" service call returns patient vital signs data collected from VistA via the Node.js web service. If the patient is currently undergoing a treatment, the vital signs returned are those taken during the treatment period. If the patient is currently not undergoing treatment, the vital signs returned are those obtained since the last treatment or all those contained within the VistA database. Data is retrieved from the COMS internal database, made available from the most recent web service call. The patient vitals service call is as follows.

- http://example.com/patient/vitals/GUID

The GUID passed to the "Patient Vitals" service call is the "PatID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a

standard Ext JS Data Store as a JSON Object based on the "PatientVitalsHistory" Model, as follows.

```
PatientVitalsHistory = [  // This is an array of individual Patient Vitals History JSON Objects
        {
                "Date" : "",                // Date the measurement was taken
                "Temp" : "",                // Temperature in degrees Fahrenheit
                "Pulse" : "",               // Pulse in beats per minute
                "BP" : "",                  // Blood Pressure taken as Systolic / Diastolic mm of Hg
                "Respiration" : "",         // Patient respirations in breaths per minute
                "Pain" : "",                // Level of pain using the standard Comparative Pain Scale
                                            // ranging from 0 - 10
                "SPO2" : "",                // Saturation of Peripheral Oxygen in the blood
                "PS" : [                    // Performance Status - Using the ECOG
                                            // (Eastern Cooperative Oncology Group) Scale
                        {
                                "PS_Level" : "",
                                "PS_Desc" : ""
                        }
                ],
                "Height" : "",                      // Patient height in inches
                "Weight" : "",                      // Patient weight in pounds
                "BSA_WeightMethod" : "",            // BSA Weight Method - derived from the allowable
                                                    // BSA weight methods table; specified when template
                                                    // applied
                "BSA_Method" : "",                  // BSA Method - derived from the allowable
                                                    // BSA methods table; specified when template applied
                "BSA_Value" : "",                   // BSA Value, calculated based on available criteria
                "BSA_Calculations" : ""             // Actual formula used to derive the current BSA Value
        }
]
```

The "PatientVitalsHistory" data is rendered in the "Patient History" Panel via an Ext JS library xTemplate.

Laboratory Results Service Call

The "Laboratory Results" service call returns laboratory results data collected from VistA via the Node.js web services. If the patient is currently undergoing a treatment, the laboratory results returned are those taken during the treatment period. If the patient is not currently undergoing a treatment, the results returned are since the last treatment or all those contained in the VistA instance. The data is then retrieved from the COMS internal database, made available from the most recent web service call. The laboratory results service call is as follows:

- http://example.com/patient/labs/GUID

The GUID passed to the "Laboratory Results" service call is the "PatID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "PatientLabs" Model, as follows.

```
PatientLabs = [          // Array of individual Laboratory Results JSON Objects
        {
                "Date" : "",            // Date of the laboratory test
                "Info" : "",            // Information on the laboratory test
                "Name" : "",            // Name of the test
                "Specimen" : "",        // Specimen used for the test
                "Result" : "",          // Results of the test
                "AccptRange" : "",      // What the acceptable range of the test is
                "InRange" : "",         // true if the results are within the acceptable range
                "Comment" : "" // Comments
        }
]
```

The "PatientLabs" data is rendered in the "Laboratory Information" panel via a Grid control from the Ext JS library. By default, data is grouped on the "Specimen" data, but may be changed and/or sorted through the Grids controls.

4.8.4. Chemotherapy Template Order Source

The "Chemotherapy Template Order Source" (CTOS) module is accessible through the "Patient" tab following patient selection and contains specific template information regarding the patient's treatment regimen.

The CTOS module contains data obtained via the Patient Template Data Service Call rendered via an Ext JS xTemplate. This information is presented in the COMS Chemotherapy Template Order Source display, as shown in **Figure 4**.

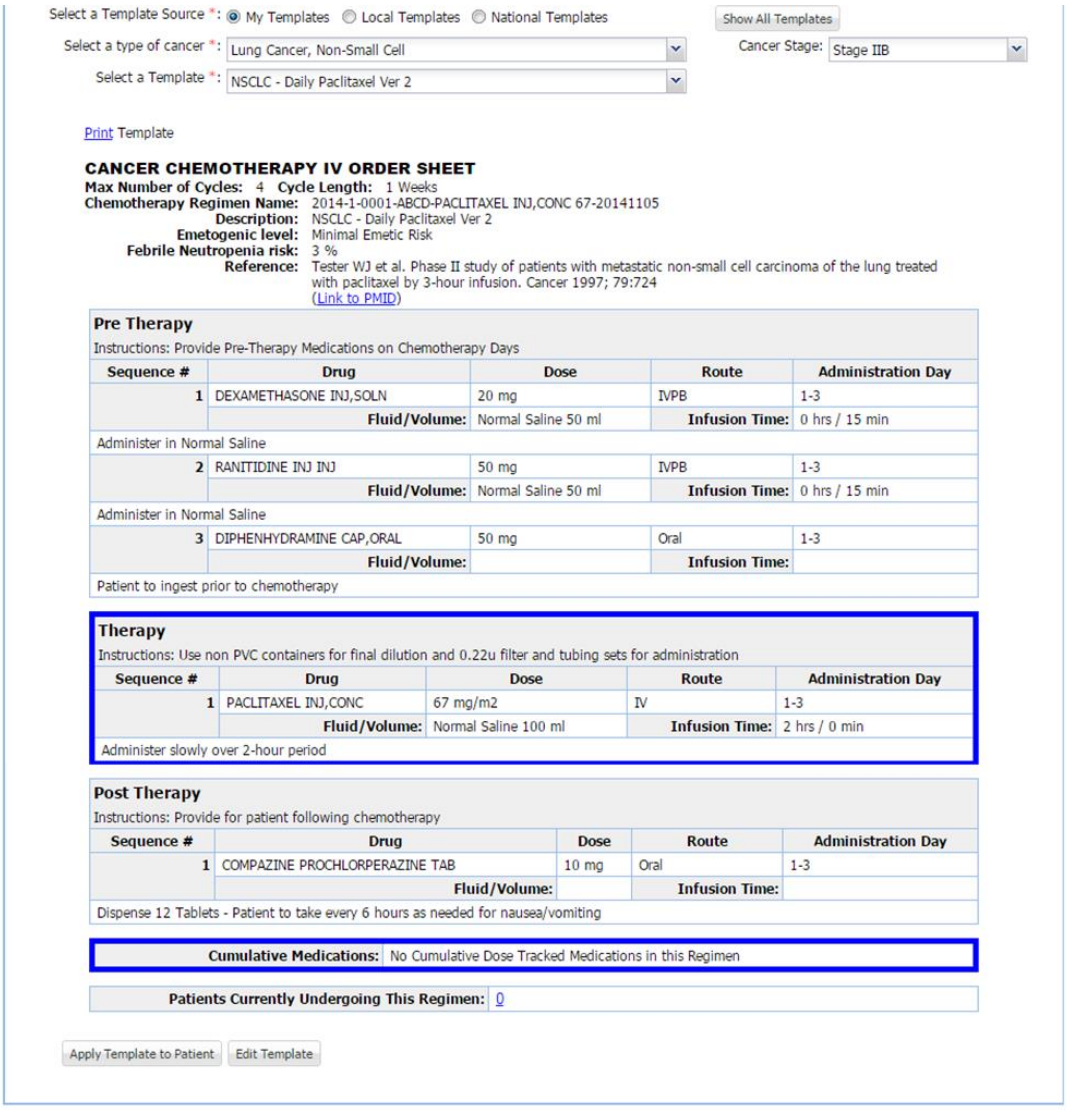

Figure 4: Chemotherapy Template Order Source Display

Template Data Service Call

The "Template Data" service call retrieves details on the specified template. Typically, this is the template currently applied to a patient, a template previously applied to a patient, or a template for review prior to applying to a patient. The template data service call is as follows:

- http://example.com/template_data/GUID

The GUID passed to the "Template Data" service call is the "TemplateID". This is obtained either from the "PatientList" service call when the patient was selected if reviewing the template currently applied to the patient, or the "Patient Treatment History" service call to review a previously applied template. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "Template Data" Model, as follows.

Template Data Model

```
{
        "id": "",
        "RegimenName": "",
        "Disease": "",
        "DiseaseStage": ""
        "CycleNumMax": "",
        "CycleLength": "",
        "CycleLengthUnit": "",

        // Emotegenic Level (Low, Med, Moderate, High)
        " ELevelName ": ""

                // Febrile Neutropenia Risk
        "FNRisk" : "",          // Pct (whole number only, 0-100)
        "FNRecom" : "",// Text for recommendation

        "References": [],

        "PreTherapy" : {},      // JSON object using the Therapy Model for the Pre Therapy
                                // portion of the treatment
        "Therapy" : {},         //  JSON object using the Therapy Model for the Therapy
                                // portion of the treatment
        "PostTherapy" : {}      //  JSON object using the Therapy Model for the Post Therapy
                                // portion of the treatment
}
```

4.8.5. Order Entry Management

The "Order Entry Management" (OEM) module is accessible through the "Patient" tab following patient selection when a template is currently applied to the selected patient. The OEM module contains data obtained via the Order Entry Management Service Call rendered via an Ext JS xTemplate, as shown in **Figure 5**.

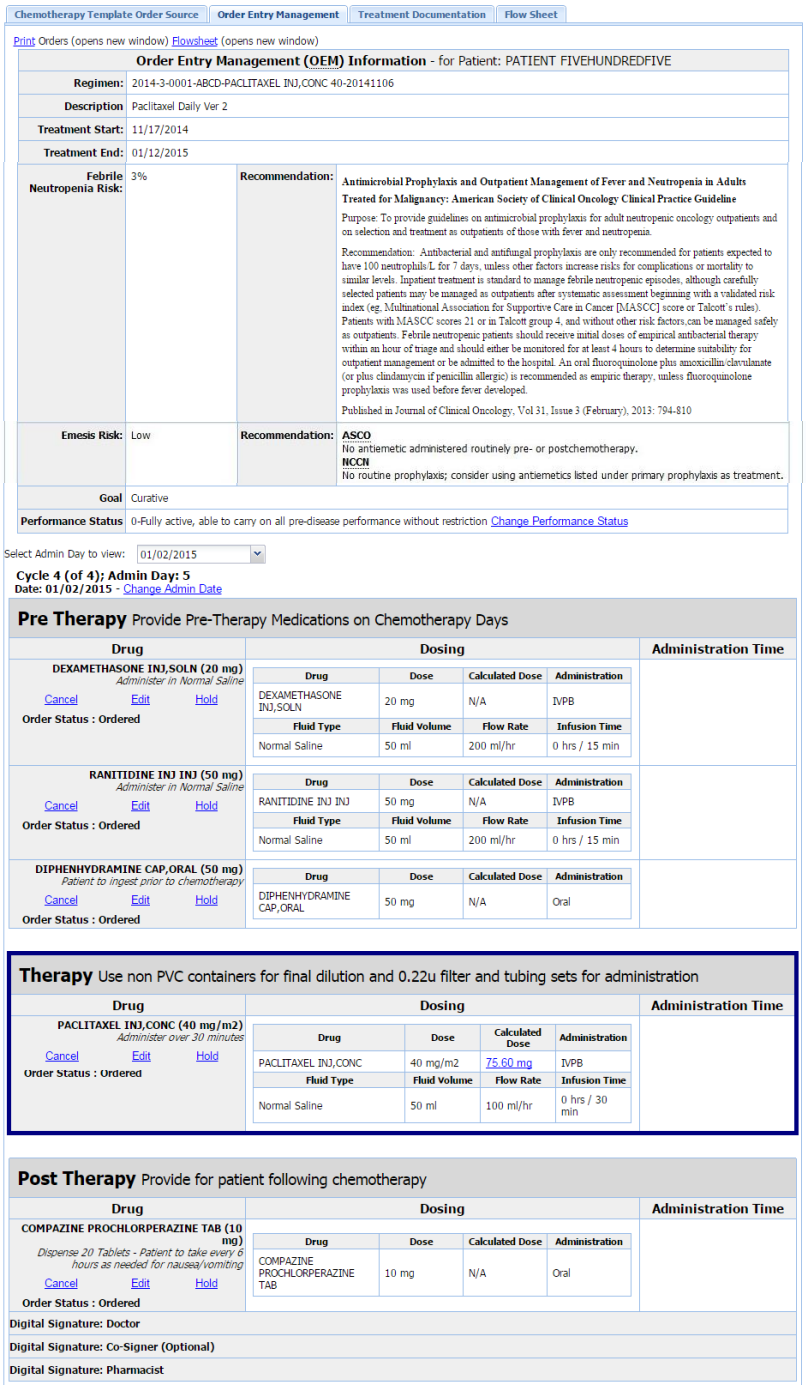

Figure 5: Order Entry Management Display

Order Entry Management Service Call

The "Order Entry Management" (OEM) service call returns all the OEM data listed in the Order Entry Management module. This information is available only if the patient is undergoing a treatment. If the patient is not currently undergoing a treatment or a request has not been made to view any historical treatment information, this service call cannot be issued. If the service call cannot be issued, the COMS application will present a "No Template has been applied to this

patient. Tab will not display" message. Figure 5 depicts the COMS Order Entry Management display for the successful return and display of the OEM service call is as follows:

- http://example.com/oem/GUID

The GUID passed to the "OEM" service call is the "OEM_ID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "OEM" Model, as follows.

OEM (Order Entry Management) Model

```
{
        "id" : "",                      // GUID for this record

        "FNRisk" : "",                  // Febrile Neutropenia Risk
                                        // Pct (number only, 0-100)
        "FNRecom" : "",                 // Text for recommendation

                                        // Emesis Risk
        "ELevelID" : "",                // GUID for the Emesis Risk record Information
        "ELevelName" : "",              // Level name (e.g. "Low", "Medium", "High", etc)
        "ELevelRecomASCO" : "",         // American Society of Clinical Oncology (ASCO)
                                        // Recommendation
        "ELevelRecomNCCN" : "",         // National Comprehensive Cancer Network (NCCN)
                                        // Recommendation
        "numCycles" : "",               // Number of treatment cycles in this template
        "DaysPerCycle" : "3",           // # of days in a treatment cycle
                                        // (Note a cycle may be measured in days, weeks,
                                        // months but this is always days)
        "Goal" : "",                    // Goal for this particular treatment; curative or
                                        // palliative (set at time template is applied to patient)
        "ClinicalTrial" : "",           // If this is a clinical trial then this is the trial description
                                        // otherwise it's blank
        "Status" : "Ordered",           // status
        "PerformanceStatus" : "",       // Performance Status, initially set at time template is
                                        // applied, but can be changed during the treatment
        "LastPSChange" : "",            // Date the Performance Status was last changed
        "OEMRecords" : []               // Array of OEM Admin Day Models, as follows:
}
```

OEM Admin Day Model

```
{
        "id" : "",              // GUID for this record
        "Cycle" : "",           // Cycle # (1-N)
        "Day" : "",             // Day in Cycle # (1-N)
        "AdminDate" : "",       // Calendar Date (mm/dd/yyyy)
        "PreTherapy" : {},      // JSON object using the Therapy Model for the Pre Therapy
```

```
                                    // portion of the treatment
        "Therapy" : {},             // JSON object using the Therapy Model for the Therapy
                                    // portion of the treatment
        "PostTherapy" : {}          // JSON object using the Therapy Model for the Post Therapy
                                    // portion of the treatment
}
```

Therapy Model

```
{
        "instr" : "",               // Instructions for this type of therapy
        "Therapy" : []              // Array of individual Therapy Order Models
}
```

Therapy Order Model

```
{
        "id" : "",                  // GUID for this record
        "Order_ID" : "",            // GUID for the Order Record linked to this record
        "Instructions" : "",        // Instructions for this particular treatment
        "Med" : "",                 // Medication for this treatment
        "MedID" : "",               // GUID for the Medication
        “Sequence” : “”,            // Order in which this medication is to be given
        "AdminTime" : "",           // When on the Admin Day the medication is to be administered
        "Dose" : "",                // Dosage to be administered
        "DoseUnits" : "",           // Units of Measure for the Dosage
        "AdminMethod" : "",         // Method of administering the medication
        "BSA_Dose" : "",            // Dosage to be administered based on Body Surface Area
        "FluidType" : "",           // Type of fluid to be used for IV administration methods
        "FluidVol" : "",            // Fluid Volume to be used for IV administration methods
        "FlowRate" : "",            // Rate of flow of fluid (plus medication) to be used for
                                    // IV administration methods
        "InfusionTime" : ""         // How long it should take to administer the medication if
                                    // delivering via IV administration methods; automatically
                                    // calculated as FluidVol / FlowRate
}
```

The “Order Entry Management” data is rendered in the “Order Entry Management” module within the “Patient Information” section of the “Patient” tab via an xTemplate component from the Ext JS library.

A Combobox control for selecting a particular Administration Day within the entire Treatment period, or “Show All” Administration Days is available for filtering the “Order Entry Management” data.

4.8.6. Treatment Documentation

The "Treatment Documentation" module set contains six activity-based panels for nurses or other healthcare professionals to enter information on patient treatment and assessment.

At the top of the "Treatment Documentation" module is the "Chemotherapy/Biotherapy" header. This header lists the current Regimen name, link to an external Flow Sheet for ready review of information, Febrile Neutropenia Level and Emetogenic Level (both with expandable views for recommendations), and Cycle, Day, and Date information, as shown in **Figure 6**. This header will display the message "Warning – this is not a scheduled Administration Day for this Regimen" and blank fields for Cycle, Day, and Date, as appropriate.

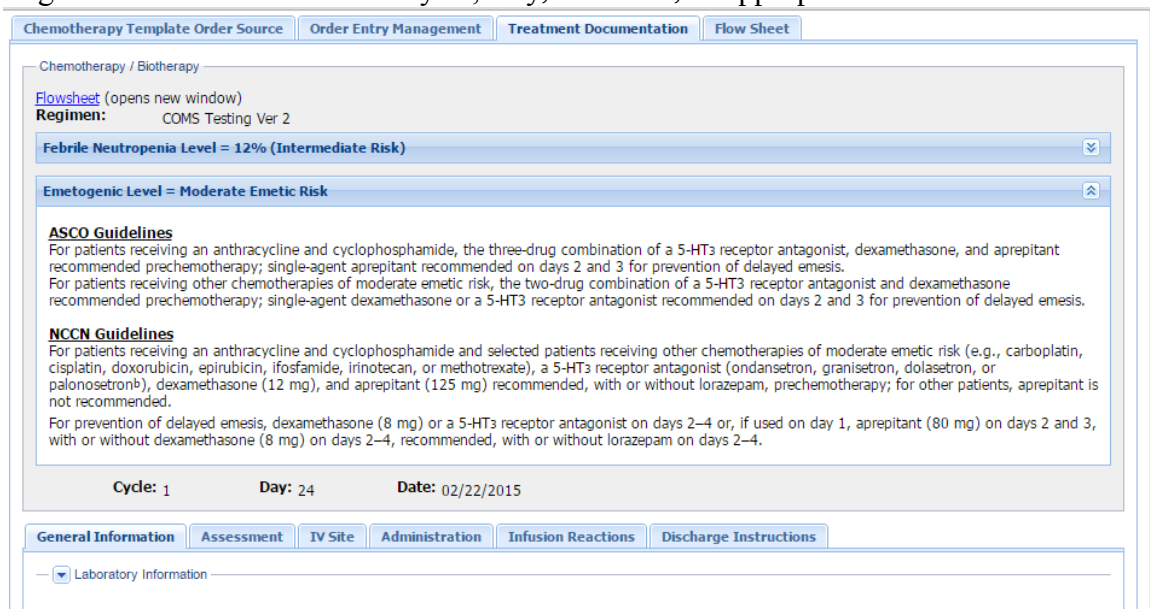

Figure 6: Treatment Documentation Header Display

Treatment Documentation – General Information Panel

The "General Information" panel of the Treatment Documentation module contains a collapsible section for displaying "Laboratory Information". This section mirrors the "Laboratory Information" section from the "Patient Information" panel, presented in the TD module for the ease of view by oncology nurses.

After "Laboratory Information", the next section is available for nurses to document "Patient Identification" and "Patient Teaching". The following section is for nursing verification, and dual verification as appropriate, of medication dosing. The remaining two sections enable nurses to record vital signs information and view historic vital signs. The TD module, General Information panel is shown in **Figure 7**.

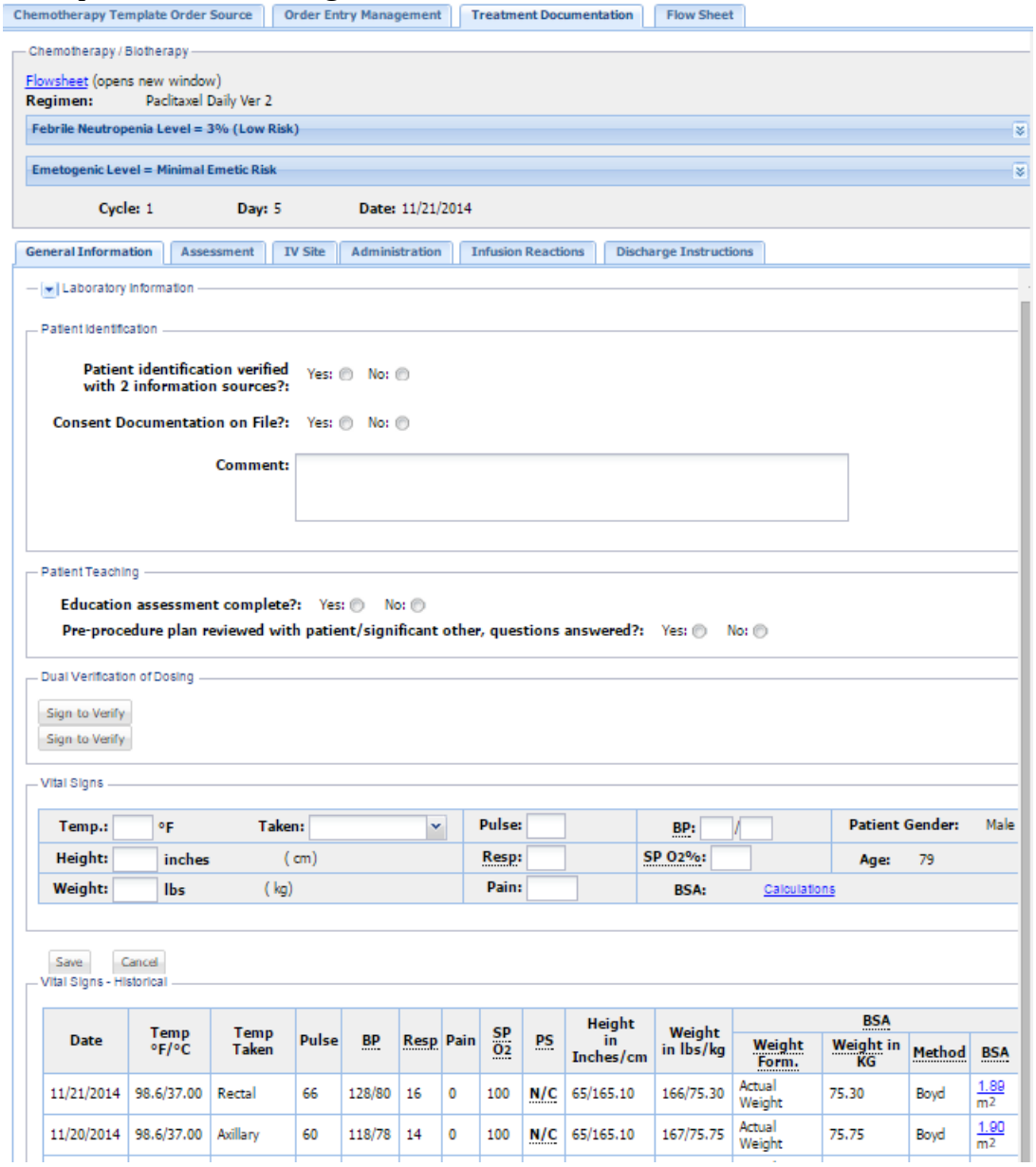

Figure 7: TD Module - General Information Panel

Oncology nurses populate the fields contained within the General Information panel. All information is then posted back to the server via the "NursingDoc GenInfo" service call when the user clicks the "Save" button. The Treatment Documentation General Information Service Call is as follows:

- http://example.com/NursingDoc/GenInfo

The "NursingDoc GenInfo" service call posts data to the COMS database in the form of a standard Ext JS Data Store as a JSON Object based on the "GenInfo" Model, as follows:

GenInfo Model

```
{
        "PatID" : "",                   // GUID to identify this specific patient
        "PatIDGood" : <BOOL>,           // true indicates that the patient ID has been verified
        "ConsentGood" : <BOOL>,         // true indicates consent obtained
        "Comment" : "",                 // Any comments entered
        "EducationGood": <BOOL>,        // true indicates Education Assessment Complete
        "PlanReviewed": <BOOL>          // true indicates Plan reviewed and all questions
                                        // answered
}
```

Upon successful completion of the "POST" request, the "NursingDoc GenInfo" service call replies back with a standard JSON object response of the form as follows:

```
{
        "success": "true",
        "message": "Gen Info Save Successful"
}
```

If the "success" parameter is false, the "message" parameter will indicate the reason for the save failure.

COMS provides two (2) buttons for "Dual Verification of Dosing". This functionality permits two (2) separate electronic signatures by two different healthcare professionals to confirm that the calculated dosing is correct.

COMS utilizes "Vital Signs" entries to post the patient's current vital signs back to the application's database. All Vital Signs are then displayed in the "Vital Signs – Historical" table; this mirrors the "Patient Vitals" panel. The vital signs entered are posted back to the server via the "Patient Vitals" service call when the user clicks the "Save" button.

Note that the "Save" button can issue two (2) posts. One post is for the "Patient Identification" and "Patient Teaching" while the other is for the "Vital Signs". If the "Patient Identification" and "Patient Teaching" or "Vital Signs" sections have not been populated, the "Save" button will only post back to the server for the populated section(s).

Treatment Documentation – Assessment Panel

The "Assessment" panel of the Treatment Documentation module contains a collapsible section for displaying "Notes on Assessment Events". This is a block of static text.

Following the "Notes on Assessment Events" section, COMS provides a section for completing the nurse's assessment of adverse events the patient experienced since the last treatment. This section consists of an interactive table to document toxicities, as loaded in Site Configuration, and identify those to trigger an Adverse Event alert. When toxicities are selected, COMS will present options to select the Common Toxicity Criteria for Adverse Events (CTCAE) grade for the event, then presents details based on the selected toxicity/grade, and displays a comment section to further qualify the assessment toxicity/adverse event. This panel is shown in **Figure 8**.

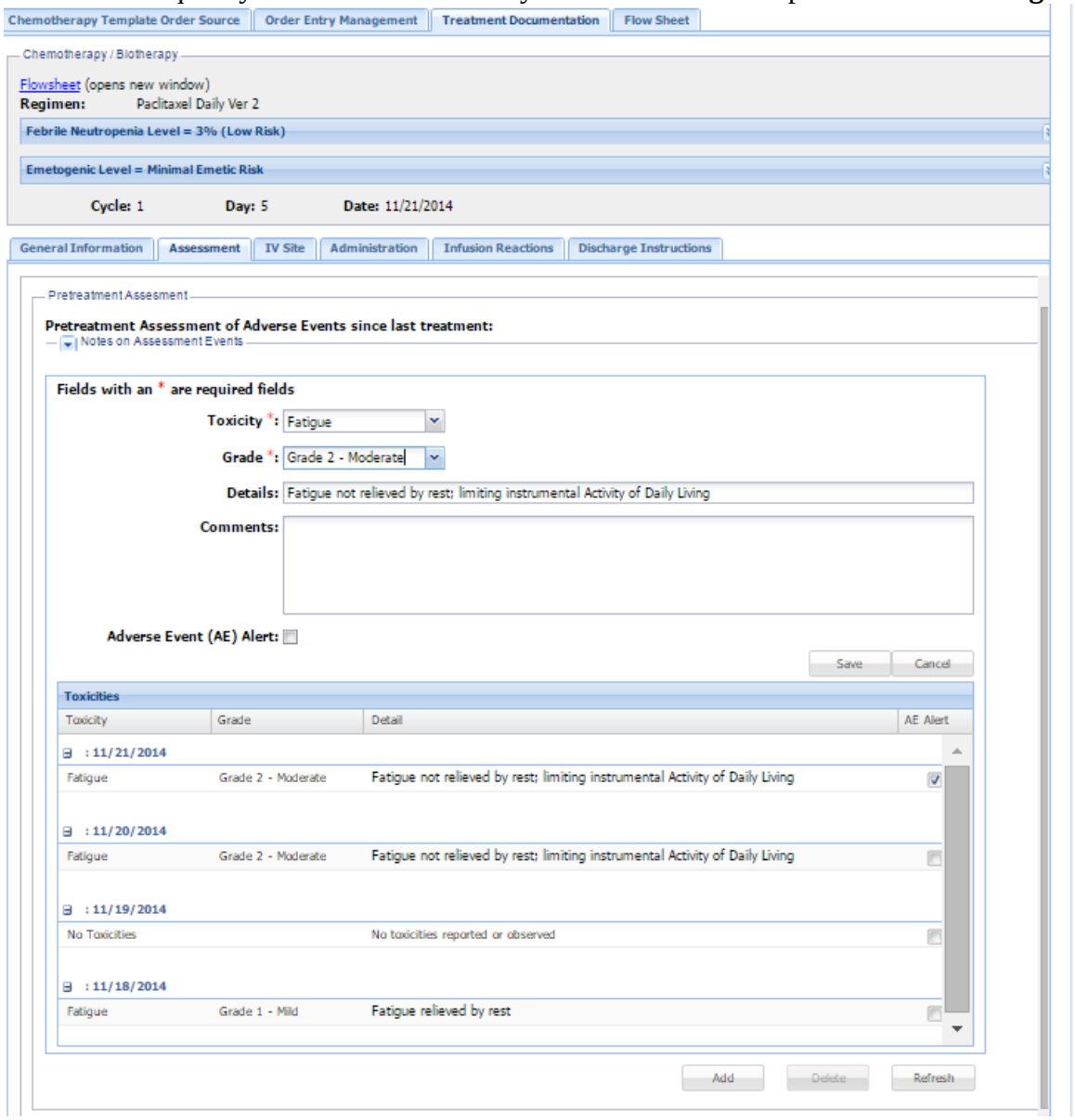

Figure 8: TD Module - Assessment Panel

The user selects the "Add" button to add assessment toxicities within the form. Upon completing the Assessment panel's toxicity form, the nurse user must click the "Save" button for the data to be posted back to the server as part of the patient's oncology record.

Populated fields within the form are posted back to the server via the "NursingDoc Assessment" service call upon clicking the "Save" button. The Treatment Documentation Assessment service call is as follows:

- http://example.com/NursingDoc/Assessment

The "NursingDoc Assessment" service call posts data in the form of a standard Ext JS Data Store as a JSON Object based on the "ND_Assessment" Model, as follows:

TD_Assessment Model

```
{
        "PatID" : "",                   // GUID to identify this specific patient
        "AssessmentDetails" : []        // Array of individual "Assessment" Model objects
}
```

Assessment Model

```
{
        "sequence",             // The order that this record is to be displayed (
                                // fatigue = 1,
                                // anorexia = 2,
                                // nausea = 3,
                                // vomiting = 4,
                                // diarrhea = 5 all others as neeeded
        "fieldLabel",           // The label for the assessment: "Fatigue", "Anorexia",
                                // Nausea", etc
        "choice",               // The value chosen for this assessment: true, false, null
        "comments",             // The user entered comments
        "levelChosen"           // The level of the Assessment: 1, 2, 3, 4 etc.
                                // based on the type of assessment
}
```

Upon successful completion of the "POST" request the "NursingDoc Assessment" service call replies back with a standard JSON object response in the following form:

```
{
        "success": "true",
        "message": " Assesment Info Save Successful"
}
```

If the "success" parameter is false, the "message" parameter will indicate the reason for the save failure.

Treatment Documentation – IV Site Panel

The "IV Site" panel of the Treatment Documentation module contains several sections for nursing users to populate, as follows:

- IV Access
- Site Appearance
- Brisk blood return verified
- Comments

The IV Site panel within the TD module is shown in **Figure 9**.

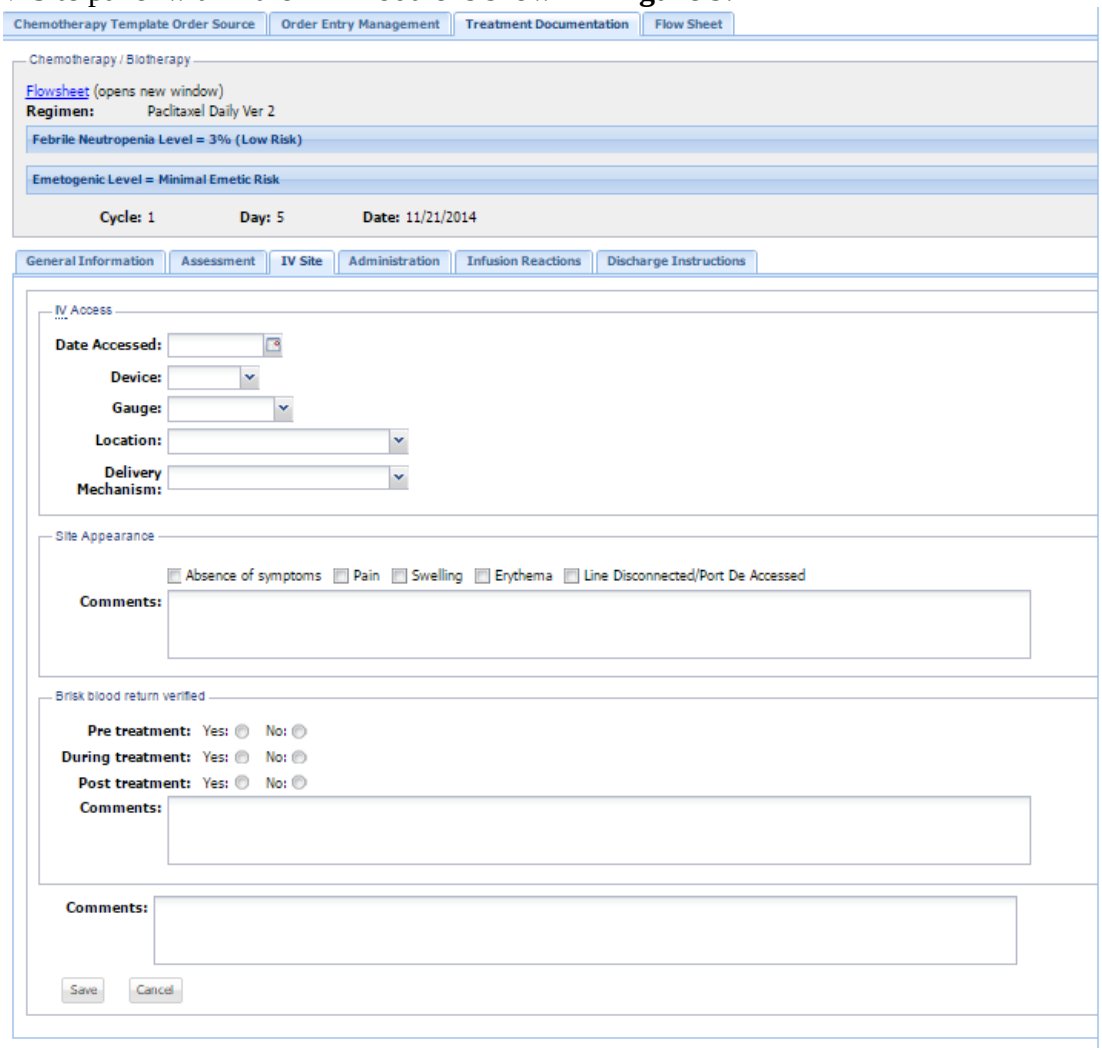

Figure 9: TD Module - IV Site Panel

Upon completing the form, the user must click the "Save" button for the data to be posted back to the server and COMS database.

Populated fields within the form are posted back to the server via the "NursingDoc IVSite" service call upon clicking the "Save" button. The Treatment Documentation IV Site service call is as follows:

- http://example.com/NursingDoc/IVSite

The "NursingDoc IVSite" service call posts data in the form of a standard Ext JS Data Store as a JSON Object based on the "ND_IVSite" Model, as follows:

TD_IVSite Model

```
{
        "PatID" : "",                       // GUID to identify this specific patient
        "DateAccessed" : "",                // Date – mm/dd/yyyy format
        "Device" : "",                      // From the "Device" Combobox
        "Gauge" : "",                       // From the "Gauge" Combobox
        "Location" : "",                    // From the "Location" Combobox
        "Appearance" : [],                  // Array of Site Appearance Check Boxes checked
        "Appearance_Comments" : "",         // Comments
        "BBR_Pre" : <BOOL>,                 // T/F for Brisk Blood Return verified before treatment
        "BBR_During" : <BOOL>,              // T/F for Brisk Blood Return verified during treatment
        "BBR_Post" : <BOOL>,                // T/F for Brisk Blood Return verified after treatment
        "BBR_Comments" : "",                // Comments
        "Gen_Comments" : ""                 // General Comments
}
```

Upon successful completion of the "POST" request, the "NursingDoc IVSite" service call replies back with a standard JSON object response in the following form:

```
{
        "success": "true",
        "message": " IV Site Info Save Successful"
}
```

If the "success" parameter is false, the "message" parameter will indicate the reason for the save failure.

Treatment Documentation – Administration Panel

As shown in **Figure 10**, the "Administration" panel of the Treatment Documentation module is a form grid for the user to document medication administration. The user confirms pre-populated entries in ordinal sequence for each medication category and, if necessary, edits a cell or selects from a cell Combobox, to accurately record the dose of medications administered. The user must populate the "Start Time" and the "End Time" for each medication then select the "Sign to Verify" link. Selection of this link will prompt the user for his/her VistA access/verify code credentials for digital signature/verification of the individual medications administered.

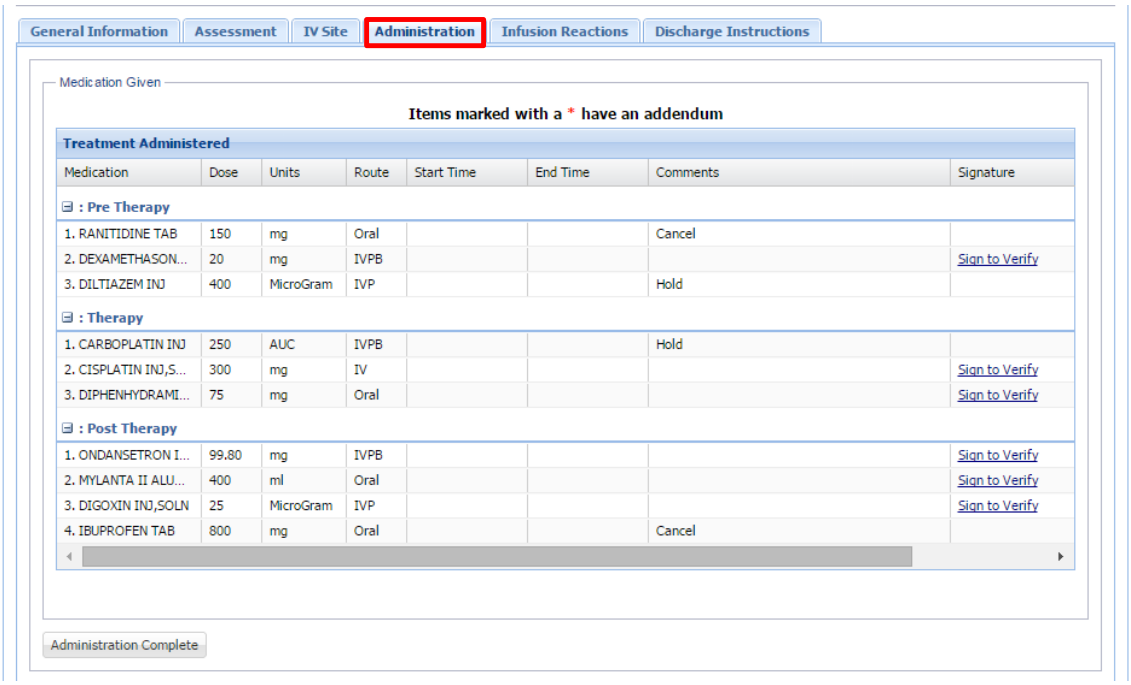

Figure 10: TD Module - Administration Panel

When the user accesses the "Administration" panel, COMS will issue an "Orders" service call to retrieve all the dispensed orders for this patient. The application applies a filter to the Ext JS Data Store for the "Orders" service call to filter medications for display. Only those medications with an "orderstatus" of "Dispensed" and "PatID" of the current selected patient are returned by the "Orders" service call for documentation on the "Administration" panel; other medications for the PatID and today's date are listed with the current status of cancel, hold, cleared, or finalized, as appropriate.

At the same time a "Treatment" service call is issued via an http "GET" request to retrieve the current treatments which have been applied to the current patient. The treatment service call is as follows:

- http://example.com/ Treatment /<GUID>

The GUID passed to the "Treatment" service call is the "PatID" obtained from the "PatientList" service call when the patient was selected. Data is returned in the form of a standard Ext JS Data Store as a JSON Object based on the "Administration" Model noted below.

Administration Model

This "Administration Model" supports both http "GET" and http "PUT" requests. Accordingly, the "GET" request will return a standard Ext JS "GET" service request as follows:

```
{
        "success" : true,
        "total" : INT - representing the # of records returned
        "records" : [] - array of 1 or more records of the "Treatment Model"
}
```

Single Treatment Model object

```
{
        "PatID" : "",               // GUID to identify this specific patient
        "PatTreatID" : "",          // Patient Treatment ID - uniquely identify all information
                                    // related to this particular treatment regimen
        "TemplateID" : "",          // GUID to identify the specific template used in this
                                    // treatment regimen
        “TreatmentID” : “”,         // GUID for this particular Treatment Record
        "Cycle" : "",               // Cycle # (1-N)
        "Day" : "",                 // Day in Cycle # (1-N)
        "AdminDate" : "",           // Calendar Date (mm/dd/yyyy)
        "Med" : "",                 // Medication given
        "MedID" : "",               // GUID for the Medication
        "StartTime" : "",           // Date/Time (mm/dd/yyyy hh:mm am/pm) the med was
                                    // administered
        “EndTime” : “”,             // Date/Time (mm/dd/yyyy hh:mm am/pm) the med was
                                    //stopped (only for IV meds)
        "Dose" : "",                // Dosage administered
        "DoseUnits" : "",           // Units of Measure for the Dosage
        "AdminMethod" : "",         // Method of administering the medication
        “Comments” : “”,            //
        “User” : “”,                // ID of the user who digitally signed this record
        “Date” : “”,                // Date/Time (mm/dd/yyyy hh:mm am/pm) of signature
        “Med_Original” : “”,        // Original values for the Med, Dose, Units and Administration
        “Dose_Original” : “”,       // method
        “Units_Original” : “”,      //
        “AdminMethod_Original” : “”,   //
}
```

When the “Treatment” service call sends data back to the server via an http “PUT” request, only a single Treatment Model record is sent. As each medication treatment is digitally signed, a “Treatment” service call is issued via an http “PUT” to post a single Administration Model record back to the VistA RX record and the supporting backend database.

Treatment Documentation – Infusion Reactions Panel

The “Infusion Reactions” panel of the Treatment Documentation module contains four (4) collapsible sections to document infusion reactions to the chemotherapy treatment. Each section has a series of check boxes. When the user checks a box, COMS will display a comment field for entering additional details of numerous selected reactions, as appropriate. When the user saves the panel, COMS presents an opportunity to identify the reaction(s) to trigger an Adverse Event alert. **Figure 11** shows the Infusion Reaction panel and identification opportunity for the four sections as follows:

- Section 1: Extravasation
- Section 2: Cytokine-Release Syndrome
- Section 3: Hypersensitivity or Anaphylaxis
- Section 4: Other

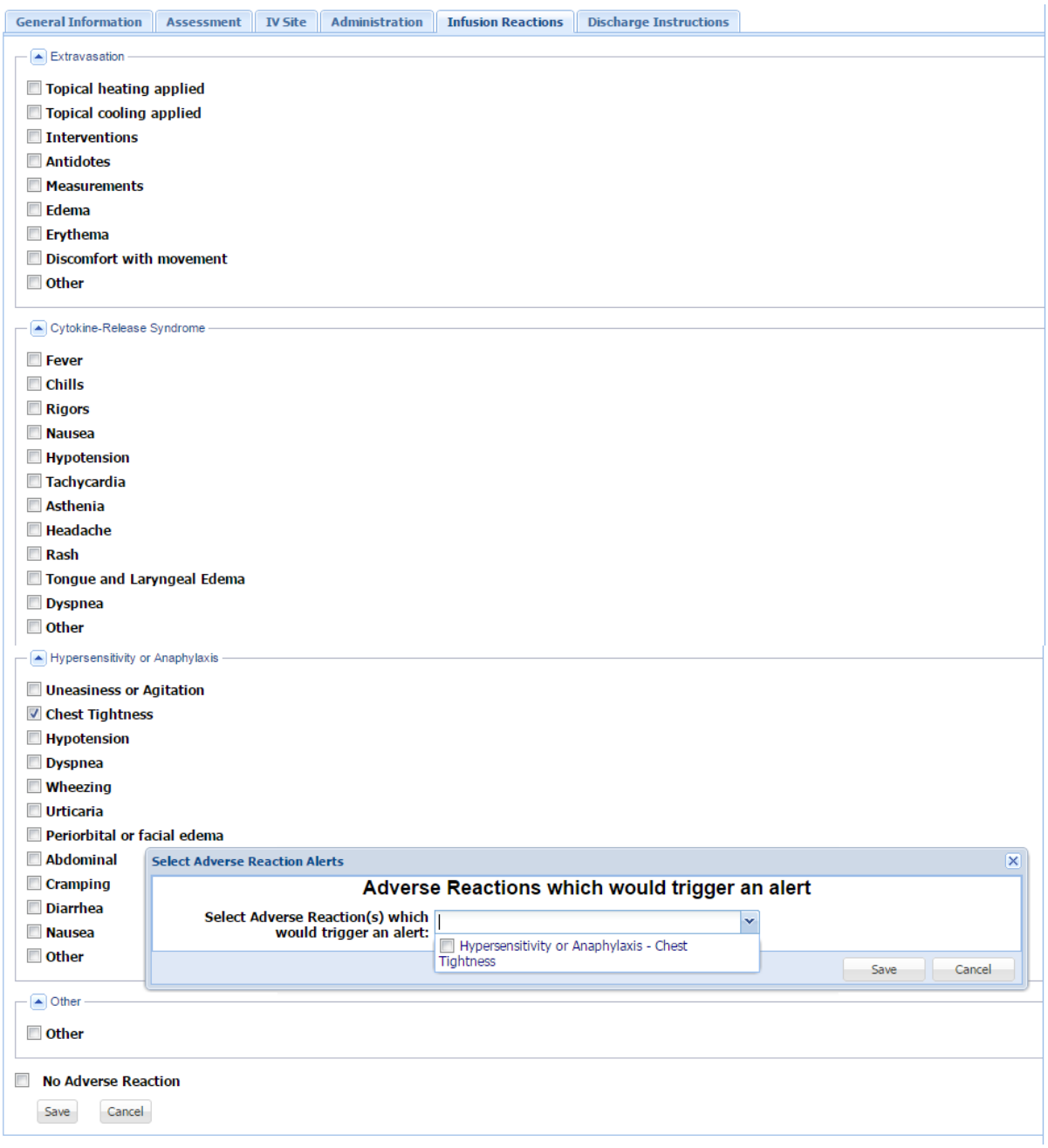

Figure 11: TD Module - Infusion Reactions Panel

The "Infusion Reactions" panel employs an "Infusion" service call for both "GET" and "POST" http requests in the following format:
http "GET" Request

- http://example.com/ Infusion/GUID

http "POST" Request

- http://example.com/ Infusion

The GUID passed to the "Infusion" service call for the http "GET" request is the "PatTreatID" obtained from the "PatientInformation" data obtained from the "Patient Data" service call when the patient was selected.

For the http "GET" request, COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the "Treatment Infusion" Model. This JSON object contains all the Infusion Reaction records for the current treatment regimen. Conversely, the http "POST" request sends a single record based on the "Infusion" Model to the server/backend database and a single "Status" message is returned upon completion of the service call to indicate the record was saved.

Treatment Infusion Model

```
{
        "success" : <BOOL>,
        "total" : <INT>,          // Representing the # of records returned
        "records" : []            // Array of 1 or more "Infusion Model" records
}
```

Infusion Model object

```
{
        "PatID" : "",             // GUID to identify this specific patient
        "PatTreatID" : "",        // Patient Treatment ID - uniquely identify all information
                                  // related to this particular treatment regimen
        "TemplateID" : "",        // GUID to identify the specific template used in this
                                  // treatment regimen
        "TreatmentID" : "",       // GUID for this particular Treatment Record
        "Cycle" : "",             // Cycle # (1-N)
        "Day" : "",               // Day in Cycle # (1-N)
        "AdminDate" : "",         // Calendar Date (mm/dd/yyyy)
         "User" : "",             // ID of the user who digitally signed this record
        "Date" : "",              // Date/Time (mm/dd/yyyy hh:mm am/pm) of signature
        "InfusionData" : []       // Array of 1 or more "Infusion Data" records
}
```

Infusion Data Model object

```
{
        "ReactionType" : "",      // The Infusion Reaction Type checked
        "Comment" : ""            // Any comments entered for this specific Infusion Reaction
                                  // Type
}
```

Upon successful completion of the "POST" request, the "Infusion" service call replies back with a standard JSON object response of the following form:

```
{
        "success": "true",
```

"message": " Infusion Reaction Info Save Successful"
}

If the “success” parameter is false, the “message” parameter will indicate the reason for the save failure.

Treatment Documentation – Discharge Instructions Panel

The “Discharge Instructions” panel of the Treatment Documentation module contains an expandable/ collapsible form for the user to enter information related to the patient’s discharge. This includes detailed information for patient education, future appointments, and discharge instructions provided to the patient. **Figure 12** shows the Discharge Instructions panel.

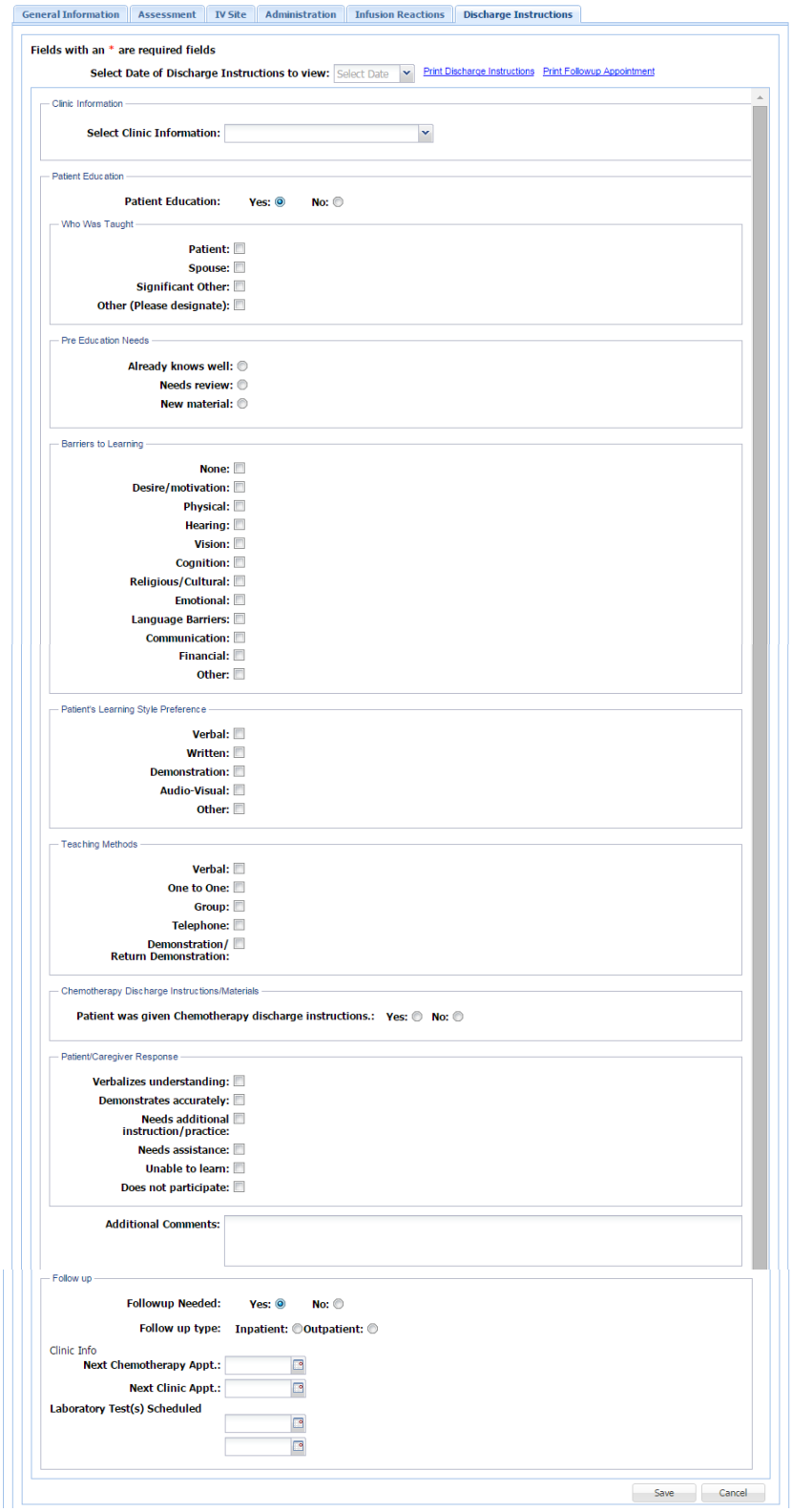

Figure 12: TD Module - Discharge Instructions Panel

The "Discharge Instructions" panel employs a "Discharge" service call. This allows for both "GET" and "POST" http requests in the following format:

http “GET” Request

- http://example.com/ Discharge/GUID

http “POST” Request

- http://example.com/ Discharge

The GUID passed to the “Discharge” service call for the http “GET” request is the “PatTreatID” obtained from the “PatientInformation” data obtained from the “Patient Data” service call when the patient was selected.

For the http “GET” request, COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the “Discharge Instructions” Model. This JSON object contains the Discharge Instructions record for the current treatment regimen. The http “POST” request sends a single record based on the “Discharge Instructions” Model to the server/backend database and a single “Status” message is returned upon completion of the service call to indicate the record was saved.

Discharge Instructions Model

```
{
        "success" : <BOOL>,
        "total" : <INT>,          // Representing the # of records returned (should == 1)
        "records" : []            // Array of 0 or 1 “Discharge” record
}
```

Discharge Model object

```
{
        "PatID" : "",             // GUID to identify this specific patient
        "PatTreatID" : "",        // Patient Treatment ID - uniquely identify all information
                                  // related to this particular treatment regimen
        "TemplateID" : "",        // GUID to identify the specific template used in this
                                  // treatment regimen
        “TreatmentID” : “”,       // GUID for this particular Treatment Record
        "Cycle" : "",             // Cycle # (1-N)
        "Day" : "",               // Day in Cycle # (1-N)
        "AdminDate" : "",         // Calendar Date (mm/dd/yyyy)
        “User” : “”,              // ID of the user who digitally signed this record
        “Date” : “”,              // Date/Time (mm/dd/yyyy hh:mm am/pm) of signature
        “Education” : <BOOL>,     // Patient Education checked
        “EduComments” : “”,       // Comments for Patient Education
        “Followup” : “”,          // Inpatient/Outpatient Followup
        “NextChemo” : “”,         // Date for next Chemo Apt (mm/dd/yyyy)
        “NextClinic” : “”,        // Date for next Clinic Apt (mm/dd/yyyy)
        “LabTest1” : “”,          // Date for Lab Tech Apt (mm/dd/yyyy)
        “LabTest2” : “”,          // Date for Lab Tech Apt (mm/dd/yyyy)
        “GivenInstructions” : <BOOL>,   // Patient was given instructions
        “Instructions” : [],      // List of instructions given
```

```
        "Comments" : ""          // Comments
}
```

Upon successful completion of the "POST" request, the "Discharge" service call replies back with a standard JSON object response of the following form:

```
{
        "success": "true",
        "message": " Discharge Instructions Info Save Successful"
}
```

If the "success" parameter is false, the "message" parameter will indicate the reason for the save failure.

4.8.7. Flow Sheet

The "Flow Sheet" (FS) module is accessible through the "Patient" tab following patient selection when a template is currently applied to the selected patient. The FS module contains data obtained via the Flow Sheet Service Call rendered via an Ext JS xTemplate, as shown in **Figure 13.**

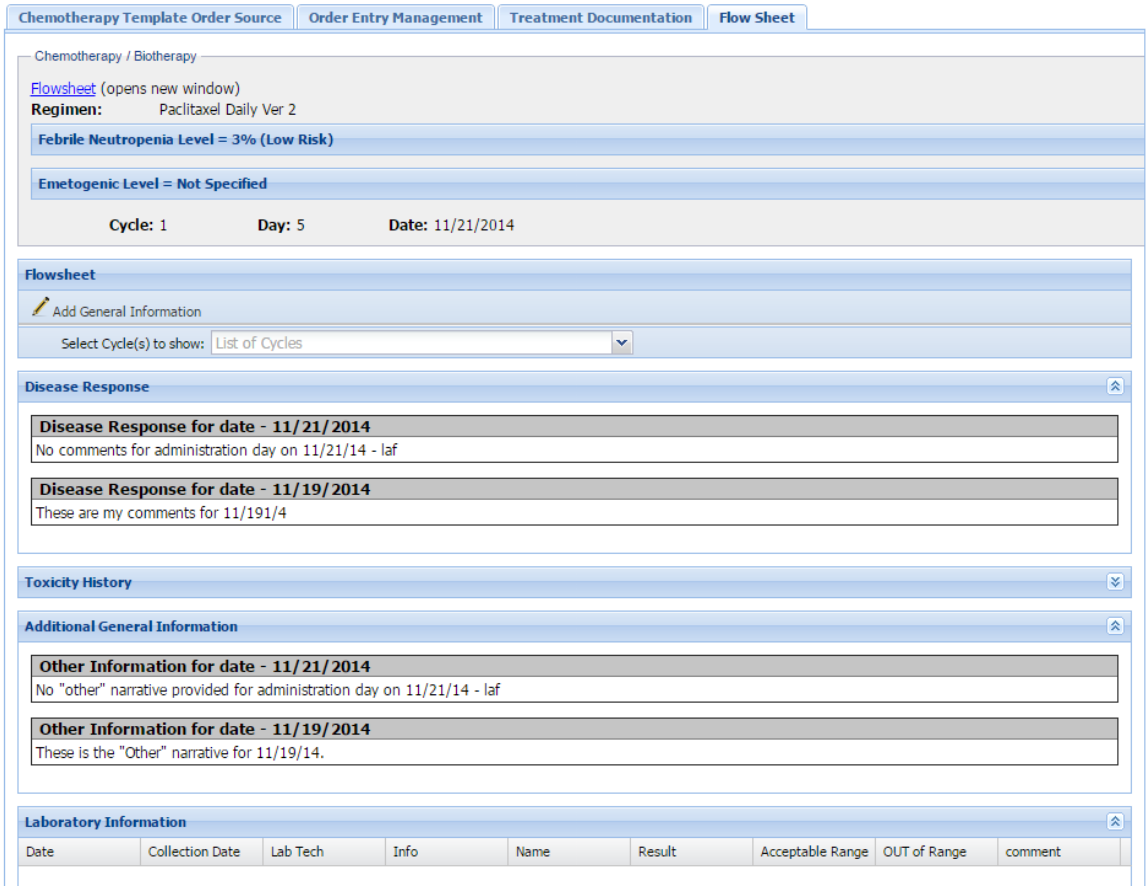

Figure 13: Flow Sheet Module Display

Within the Flow Sheet, users may select the "Add General Information" button to add clinical observations for Disease Response, Toxicities, and Other sections, as shown in **Figure 14**.

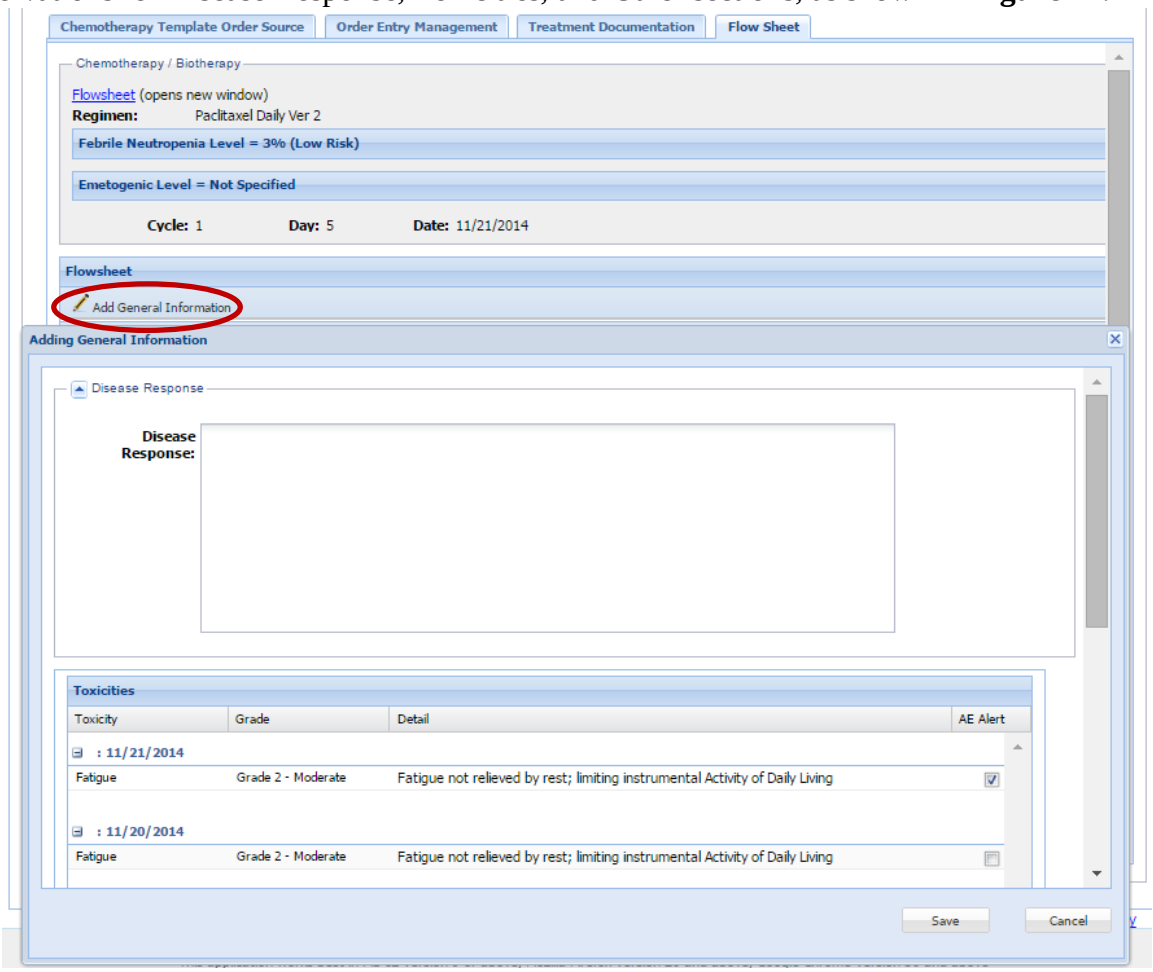

Figure 14: Flow Sheet General Information Worksheet

Users may view saved General Information entries in the expandable/collapsible panels for Disease Response, Toxicities, and Additional General Information, respectively. These panels and the Laboratory panel are shown in Figure 13 on the previous page.

4.8.8. Treatment Regimens & Summaries / End of Treatment Summary Module

The Treatment Regimens & Summaries Panel provides all functionality for the "End of Treatment Summary" (EoTS) module through the following links:

- Show Details
- Stop Treatment
- Generate End of Treatment Summary (only for historical templates)
- Show End of Treatment Summary (only for historical templates with completed summary)

The "End of Treatment Summary" (EoTS) module is accessible through the "Patient" tab following patient selection. The EoTS module contains data obtained via the End of Treatment Summary Service Call rendered via an Ext JS xTemplate. This expandable/collapsible panel is depicted in **Figure 15**.

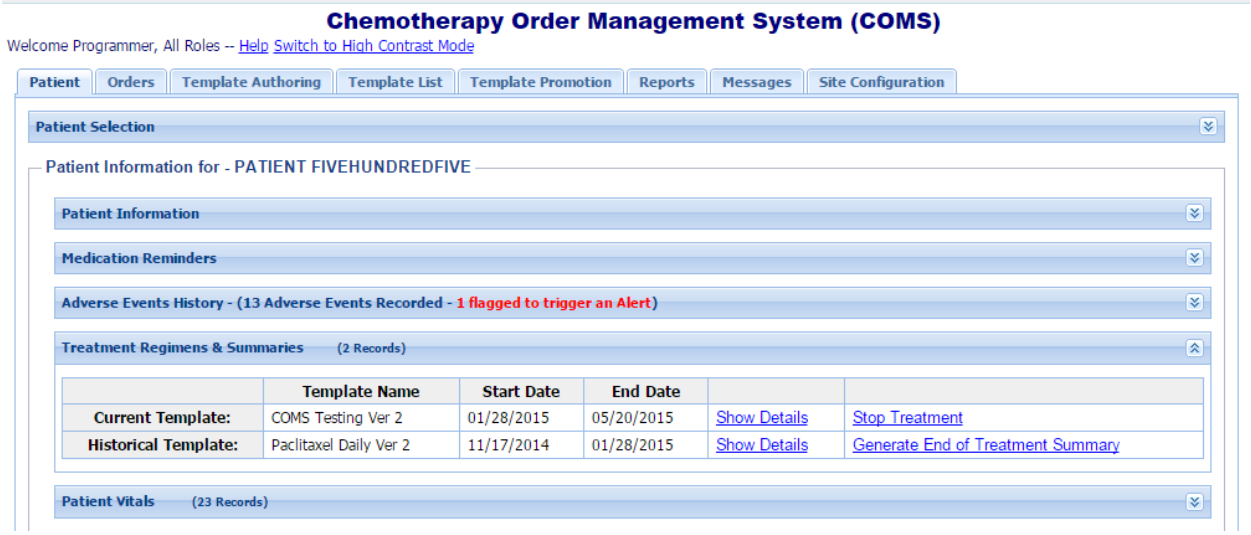

Figure 15: Treatment Regimens & Summaries Display

Show Details: Selection will display the details of the Template and treatment provided information in the following COMS clinical modules within the Treatment Details panel:

Chemotherapy Template Order Source – Template Details
Order Entry Management – Individual orders for treatment
Treatment Documentation – Details of the treatment provided and patient assessment
Flowsheet – High level overview of treatment performed and snapshot of ongoing care

Stop Treatment: Available for the Current Template only, selection will prompt the user to confirm that the treatment the patient is currently undergoing is to be stopped. If the user confirms stoppage, COMS will flag the treatment as stopped, and this link will be removed.

Generate End of Treatment Summary: Available when a treatment has ended but an End of Treatment Summary (EoTS) has not been generated, selection of this link will initiate the EoTS activity and guide the user through the process of generating an End of Treatment Summary. Users are required to specify a reason for generating an End of Treatment Summary, as shown in **Figure 16**.

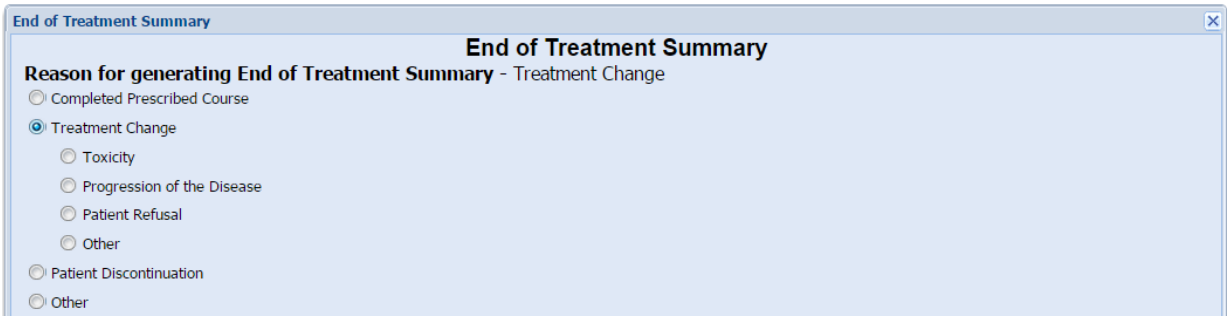

Figure 16: Reason for Generating an End of Treatment Summary

After specifying the reason, users are presented with the End of Treatment Summary worksheet for the following actions:

- View pre-populated components for patient and regimen details, type of cancer(s), and amputation(s)
- Review initial and final vital signs information for the regimen, body surface area factors and values, clinical trial information, allergy, and performance status
- View medications administered to the patient throughout the regimen; review patient disease response, toxicity side effects, and other comments as annotated on the Flow Sheet throughout the treatment regimen
- Compose a report and specify follow-up appointments for the selected patient via free text comment fields with formatting and expanding capability to accommodate lengthy narratives.

This worksheet is depicted in **Figure 17**.

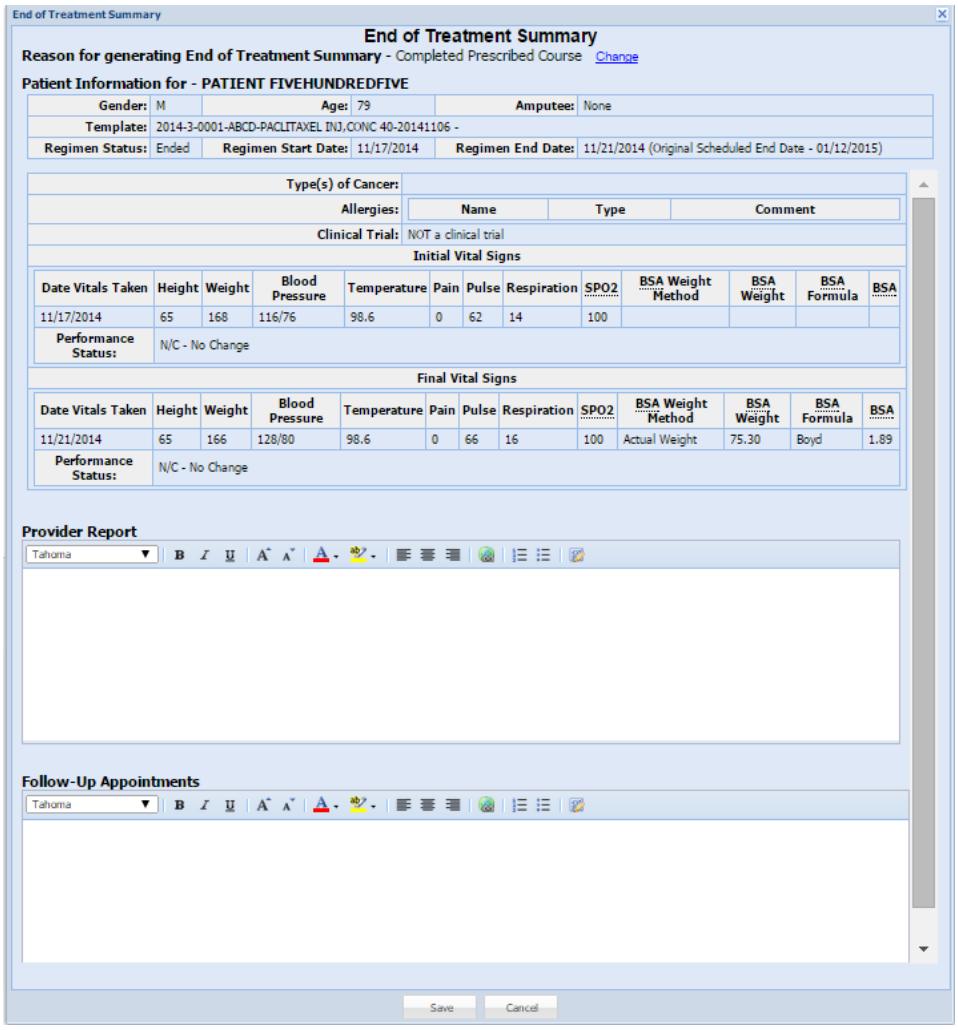

Figure 17: End of Treatment Summary Worksheet

Show End of Treatment Summary: Available when an End of Treatment Summary has been generated, selection of this link will display a read-only version of the End of Treatment Summary in a popup window. A completed End of Treatment Summary is shown in **Figure 18**.

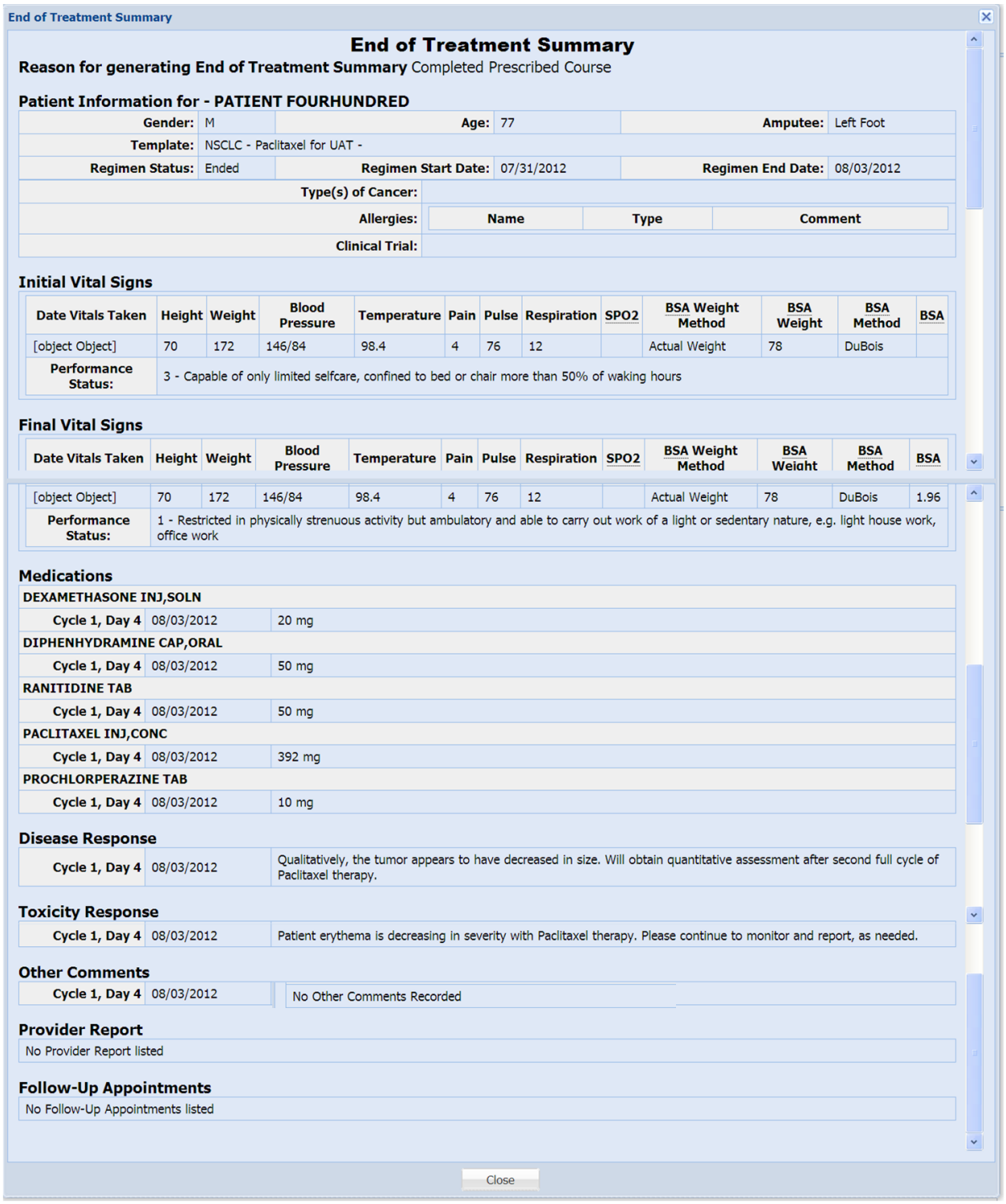

Figure 18: Completed End of Treatment Summary

4.8.9. Miscellaneous Functionality

As the underpinning of capabilities within the five clinical modules, COMS utilizes complementary functionality throughout the application. Collectively known as "Miscellaneous Functionality", these capabilities combine to enable successful interoperability among the clinical modules within COMS and with VA's electronic health record applications of VistA and CPRS. Displayed in various tabs and panels, miscellaneous functionality is presented in the next several sections.

4.8.10. Orders Tab

COMS utilizes the "Orders" tab to display all outstanding pharmacy orders for the next three (3) calendar days, as shown in **Figure 19**. Throughout the continuum of treatment, these orders will have an order status of ordered, cleared, finalized, in-coordination, dispensed, or administered for each medication prescribed by the oncology provider for the specified patient; orders may also be cancelled or held through the OEM module. COMS groups the Orders Tab by collapsible/expandable patient fields then administration dates within the respective patient grouping.

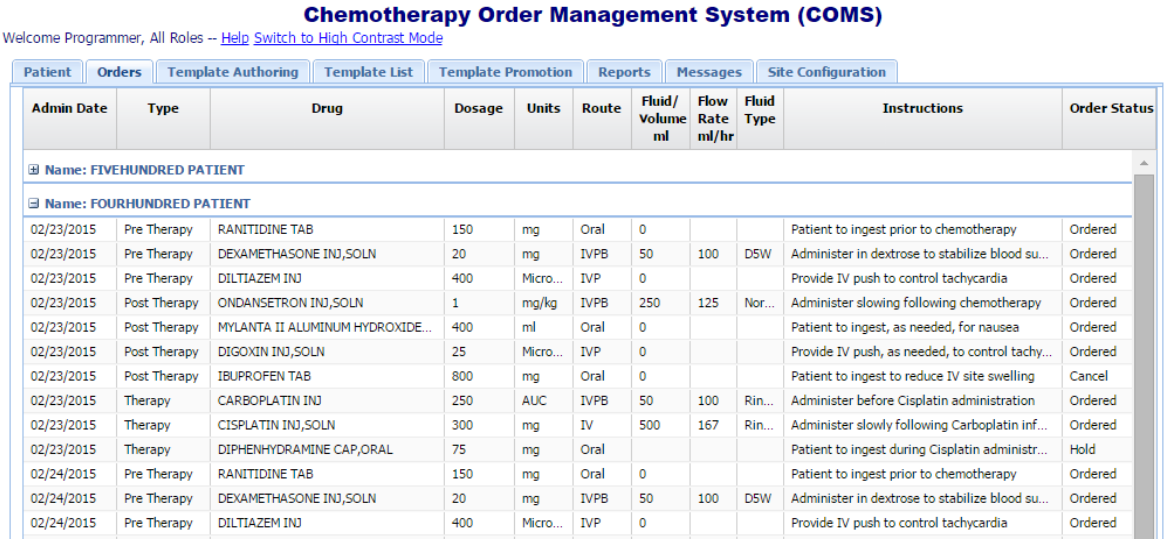

Figure 19: Orders Tab Display

Orders Service Call

The "Orders" service call returns all pharmacy order data for the next three (3) calendar days, collected from VistA via the web services within the COMS application.

- http://example.com/Orders

COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the "PharmacyOrder" Model as follows:

```
PharmacyOrder = [                    // Array of individual Pharmacy Order JSON Objects
```

```
{
    "PatID" : "",              // Patient ID - GUID from DB Table
    "Last_Name" :  "",
    "Cycle" :  ,
    "templateID" :  "",        // GUID
    "adminDay" :  ,
    "adminDate" :  "",         // mm/dd/yyyy
    "drug" : "",
  "type" :  "",
  "typeOrder" :  1,
  "dose" :  20,
  "unit" : "",
  "route" : "",
  "fluidVol" : 0,
  "flowRate": "",
  "instructions" : "",
  "Order_ID" : "",             // GUID
  "orderstatus" : "",
  "orderid" : ""               // GUID
}
```

The "Pharmacy Order" data is rendered in the "Orders" Tab via a Grid control from the Ext JS library. By default, the data is grouped by patient then the "Administration Date" data. However, users may change and/or sort the data through the Grids controls.

4.8.11. Template Authoring Tab

COMS provides the "Template Authoring" tab for authorized users to create new Chemotherapy Order Templates either from an existing template or creating an original one. **Figure 20** shows the Template Authoring tab to permit identification of regimen details, medications, references, reminders, and user-friendly name.

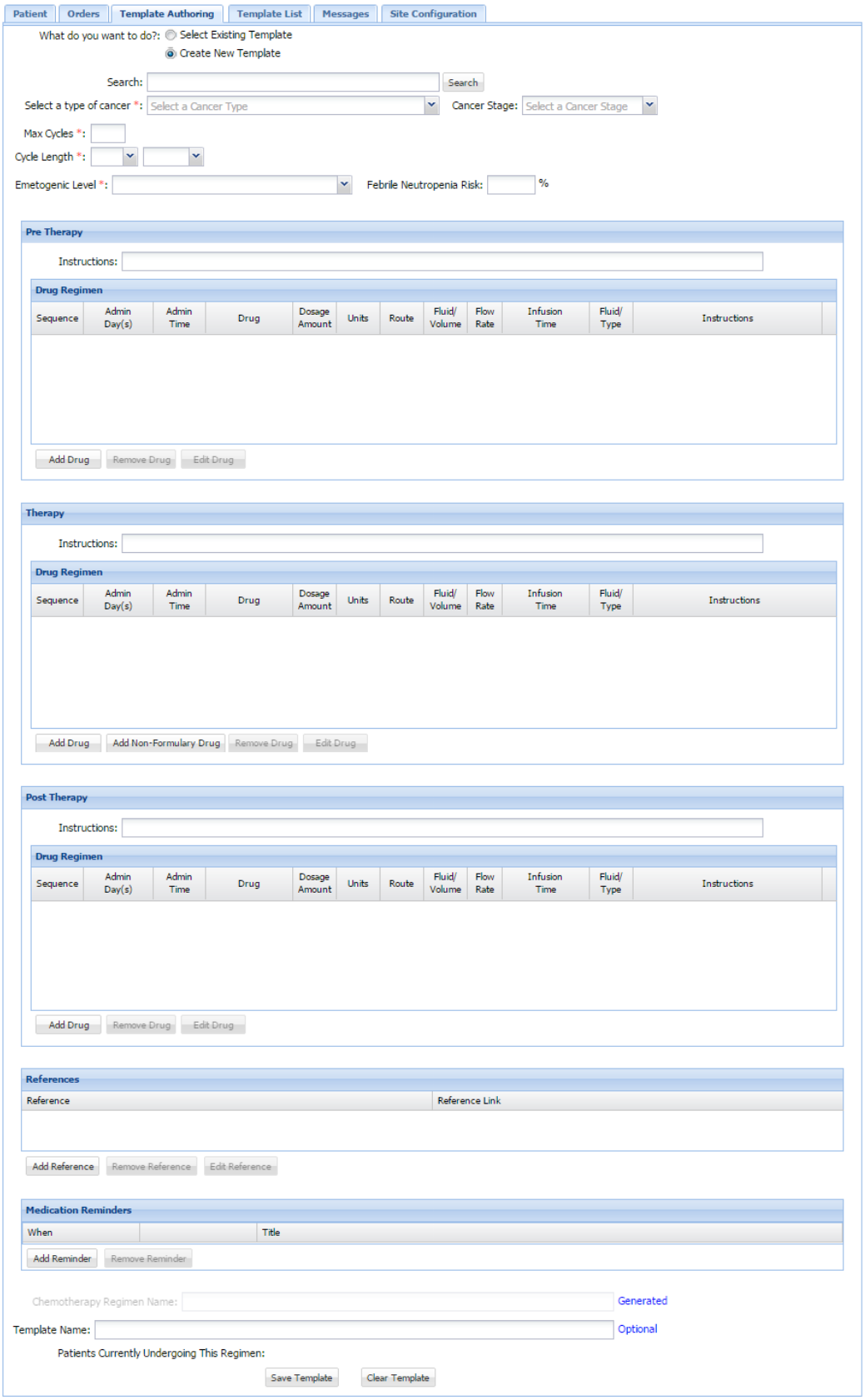

Figure 20: Template Authoring Display

Disease Selection

The "Disease Selection" Combobox is used to select the type of cancer for treatment by a specific regimen. This Combobox is linked to the "Stage Selection" Combobox to select a particular stage (or progression) of the specified cancer. Accordingly, the "Stage Selection" Combobox is dependent upon the type of cancer selected in the "Disease Selection" Combobox.

Disease Type Service Call

The "Disease Type" service call is used to obtain the list of cancers and associated stages available in the application. COMS utilizes this service call to populate the data store for the "Disease Selection" Combobox as follows:

- http://example.com/LookUp/view/DiseaseType

Data is returned in the form of a standard Ext JS Data Store as a JSON object based on the following "DiseaseType" Model:

```
{
    "id" : "",
    "name" : "",              // Name of this form of cancer
    "description" : "",       // Description of this form of cancer
    "stages" : [""]           // List of stages for this particular cancer, used as a feed for the
                              // "Stage Selection" Combobox
}
```

Emetogenic Level Selection

The "Emetogenic Level" service call is used to obtain the list of Emetogenic levels (i.e. capacity to induce emesis or vomiting). It is used to populate the data store for the "Emetogenic Level Selection" Combobox as follows:

- http://example.com/LookUp/view/Emetogenic

COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the following "EmetogenicLevel" Model:

```
{
   "id": "",                  // GUID
   "name": "",                // Level name ("Minimal" – "High")
   "level": "",               // Emetogenic Level (level 1 – level 4)
   "description" : ""         // Description for the level
}
```

Emetogenic Level Reference Pages

More information on Emetogenic Levels is available on the following clinical websites:
http://theoncologist.alphamedpress.org/content/4/3/191.full

http://www.healthprolink.com/mobile/tools/Oncology/Emetogenicscale.htm

Emesis level predicts the percentage of patients who experience emesis (i.e. vomiting) without administration of an effective emesis prophylaxis.

- 1 = < 10%
- 2 = 10-30%
- 3 = 30-60%
- 4 = 60-90%
- 5 = > 90%

http://emedicine.medscape.com/article/1355706-treatment

Further, emesis levels have defined descriptions to qualify the numeric values as the following paragraph notes: Level (Risk) : (Desc)

- Level 4 (High): More than 90% of patients who receive these chemotherapy agents experience nausea and vomiting. Carmustine, cisplatin, cyclophosphamide (>1500 mg/m2), dacarbazine, dactinomycin, mechlorethamine, streptozotocin Serotonin-receptor antagonist, dexamethasone, and aprepitant

- Level 3 (Moderate): Nausea and vomiting occurs in 30-90% of patients who receive these chemotherapy agents. Carboplatin, cyclophosphamide (< 1500 mg/m2), cytarabine (>1 g/m2), daunorubicin, doxorubicin, epirubicin, idarubicin, ifosfamide, irinotecan, oxaliplatin Serotonin-receptor antagonist, and dexamethasone

- Level 2 (Low): Nausea and vomiting occurs in 10-30% of patients who receive these chemotherapy agents. Bortezomib, cetuximab, cytarabine (< 1 g/m2), docetaxel, etoposide, fluorouracil, gemcitabine, methotrexate, mitomycin, mitoxantrone, paclitaxel, pemetrexed, topotecan, and trastuzumab Serotonin-receptor antagonist

- Level 1 (Minimal): Less than 10% of patients who receive these chemotherapy agents experience nausea and vomiting. Bevacizumab, bleomycin, busulfan, 2-chlorodeoxyadenosine, fludarabine, rituximab, vinblastine, vincristine, and vinorelbine. No antiemetic routinely administered

http://www.cancer.gov/cancertopics/pdq/supportivecare/nausea/HealthProfessional/Table3

Clinical guidelines also provide Antiemetic Recommendations by Emetic Risk Categories.

Emetic Risk Category ASCO Guidelines NCCN Guidelines

- High (>90%) risk

ASCO Guidelines - Three-drug combination of a 5-HT3 receptor antagonist, dexamethasone, and aprepitant is recommended as pre-chemotherapy treatment.

For patients receiving cisplatin and all other agents of high emetic risk, the two-drug combination of dexamethasone and aprepitant is recommended for prevention of delayed emesis.

NCCN Guidelines –Pre-chemotherapy, a 5-HT3 receptor antagonist (ondansetron, granisetron, dolasetron, or palonosetronb), dexamethasone (12 mg), and aprepitant (125 mg) recommended, with or without lorazepam.

For prevention of delayed emesis, dexamethasone (8 mg) on days 2 - 4 plus aprepitant (80 mg) on days 2 and 3 recommended, with or without lorazepam on days 2 - 4.

- Moderate (30%-90%) risk

ASCO Guidelines - For patients receiving an anthracycline and cyclophosphamide, the three-drug combination of a 5-HT3 receptor antagonist, dexamethasone, and aprepitant recommended prechemotherapy; single-agent aprepitant recommended on days 2 and 3 for prevention of delayed emesis. For patients receiving other chemotherapies of moderate emetic risk, the two-drug combination of a 5-HT3 receptor antagonist and dexamethasone recommended prechemotherapy; single-agent dexamethasone or a 5-HT3 receptor antagonist recommended on days 2 and 3 for prevention of delayed emesis.

NCCN Guidelines - For patients receiving an anthracycline and cyclophosphamide and selected patients receiving other chemotherapies of moderate emetic risk (e.g., carboplatin, cisplatin, doxorubicin, epirubicin, ifosfamide, irinotecan, or methotrexate), a 5-HT3 receptor antagonist (ondansetron, granisetron, dolasetron, or palonosetronb), dexamethasone (12 mg), and aprepitant (125 mg) recommended, with or without lorazepam, prechemotherapy; for other patients, aprepitant is not recommended. For prevention of delayed emesis, dexamethasone (8 mg) or a 5-HT3 receptor antagonist on days 2 - 4 or, if used on day 1, aprepitant (80 mg) on days 2 and 3, with or without dexamethasone (8 mg) on days 2 - 4, recommended, with or without lorazepam on days 2 - 4.

- Low (10%-30%) risk

ASCO Guidelines - Dexamethasone (8 mg) recommended; no routine preventive use of antiemetics for delayed emesis recommended.

NCCN Guidelines - Metoclopramide, with or without diphenhydramine; dexamethasone (12 mg); or prochlorperazine recommended, with or without lorazepam.

Minimal (<10%) risk -
ASCO Guidelines - No antiemetic administered routinely pre- or postchemotherapy.

NCCN Guidelines - No routine prophylaxis; consider using antiemetics listed under primary prophylaxis as treatment.

Adding a new Medication to a Therapy Treatment

The "Add New Drug Regimen" pop-up dialog box is a form used to create a new Drug Regimen record for pre-therapy, therapy, or post-therapy administration. **Figure 21** shows this pop-up dialog box.

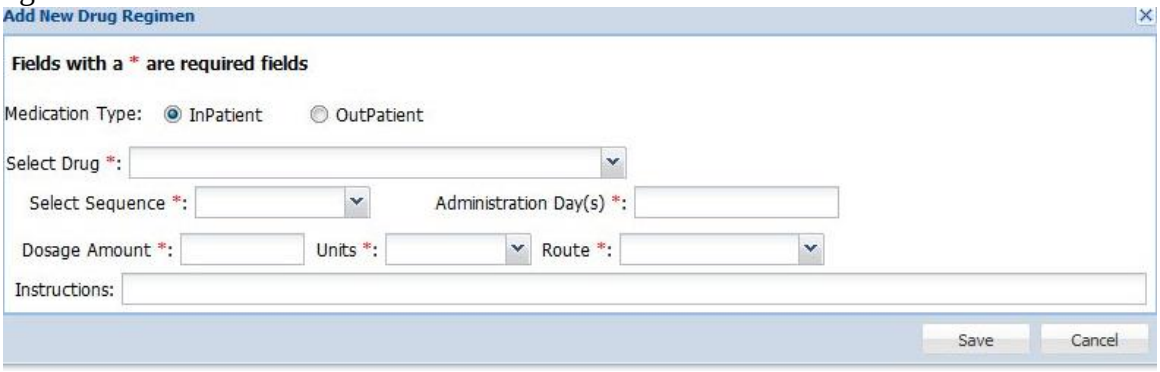

Figure 21: Add New Drug Regimen

When the user completes this form, COMS issues an http "POST" request to push the data.

Med Regimen Service Call

- http://example.com/MedRegimen

COMS passes this "MedRegimen" service call via a standard Ext JS Data record as a JSON object based on the following "MedRegimen" Model:

MedRegimen Model

```
{
        "id" : "",                  // GUID for this record
        "MedType" : "",             // Type of medication (inpatient or outpatient)
        "AdminDays" : "",           // List of days this medication is to be administered
                                    // given as a single #,
                                    // comma separated list of #'s
                                    // or a range of #'s
        "Instructions" : "",        // Instructions for this particular treatment
        "Med" : "",                 // Medication for this treatment
        "MedID" : "",               // GUID for the Medication (is this needed in this model?)
        "Sequence" : "",            // Order in which this med is to be given
        "AdminTime" : "",           // When on the Admin Day the med is to be administered
        "Dose" : "",                // Dosage to be administered
        "DoseUnits" : "",           // Units of Measure for the Dosage
        "AdminMethod" : "",         // Method of administering the medication
        "FluidType" : "",           // Type of fluid to be used for IV administration methods
        "FluidVol" : "",            // Fluid Volume to be used for IV administration methods
        "FlowRate" : "",            // Rate of flow of fluid (plus medication) to be used for
                                    // IV administration methods
        "InfusionTime" : ""         // How long it should take to administer the medication if
                                    // delivering via IV administration methods
}
```

4.8.12. Template List Tab

The Template List Tab enables authorized users to view/print all templates available within the COMS application. Grouped by type of cancer, the Template List displays both the COMS-generated and user-friendly name established during template authoring, number of patients currently undergoing treatment with the template regimen, and view/print options. The Template List Tab is shown in **Figure 22**.

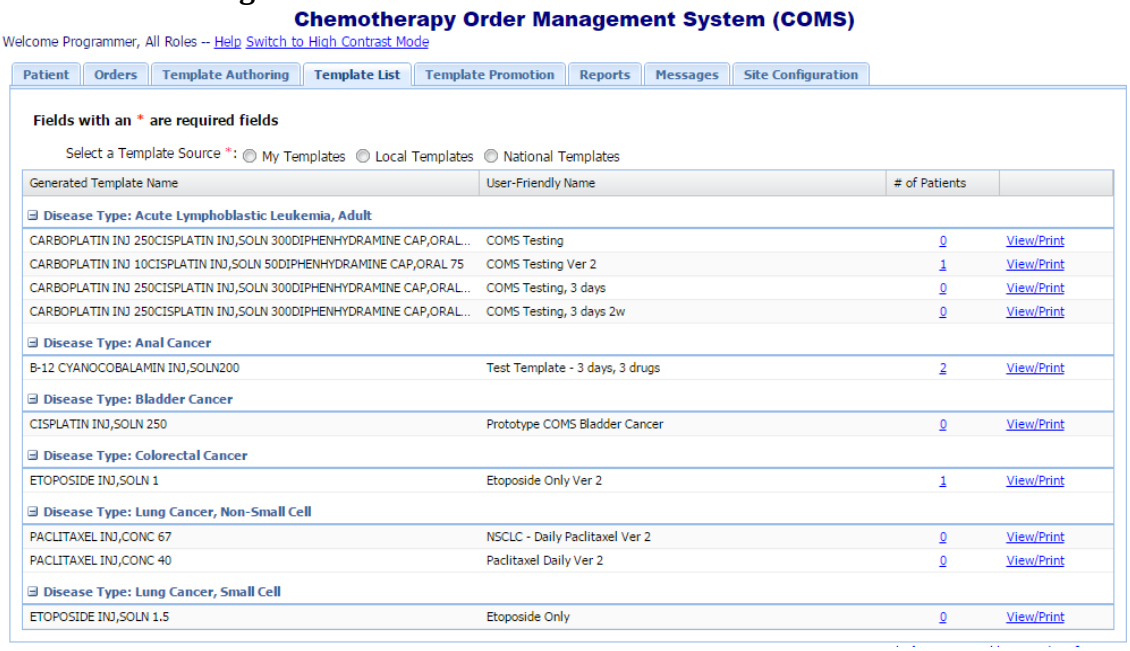

Figure 22: Template List Tab Display

4.8.13. Template Promotion Tab

The Template Promotion Tab provides super users – serving as local template managers at each facility – functionality to advance templates from single provider use to local use and local use to national consideration following the local and national vetting processes, respectively. The Template Promotion Tab displays information grouped by type of cancer with the COMS-generated and user-friendly name, location/template source (i.e. My Templates, Local Templates, or National Templates), number of patients currently undergoing treatment with the template, and view/print options, as shown in **Figure 23.**

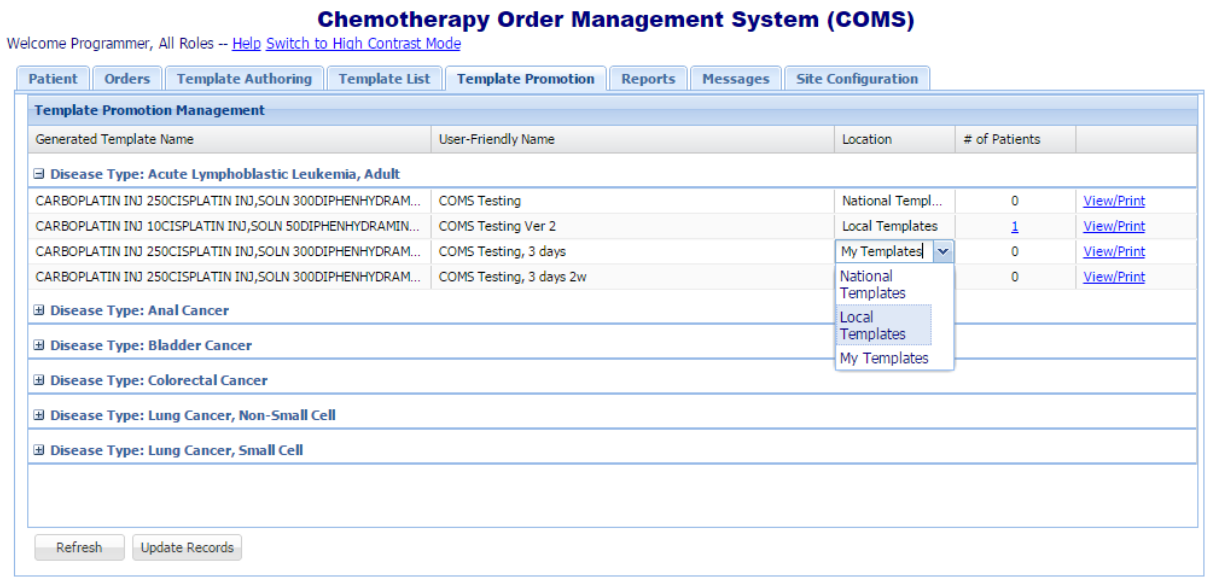

Figure 23: Template Promotion Tab Display

The Template Promotion Tab enables local template managers to designate locally created templates for review and consideration for inclusion in the central library. In this capacity, users may access this functionality in conjunction with Import/Export Templates.

4.8.14. Reports Tab

The Reports panel presents three subordinate panels for COMS administrators to configure canned reporting functionalities within the application. **Figure 24** shows the Reports Tab.

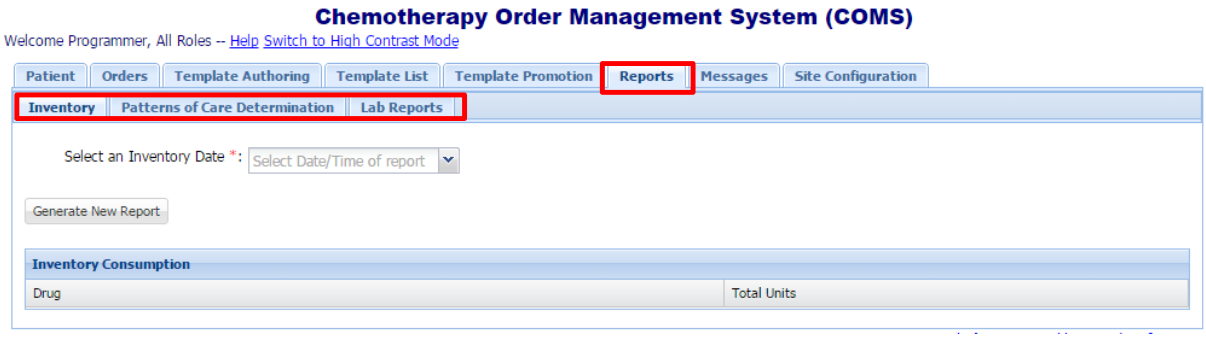

Figure 24: Reports Tab Display

Inventory

Functionality for Inventory facilitates authorized users to specify date ranges for inventory reports to aid in replenishment decisions for chemotherapy agents and related items.

Patterns of Care Determination

Patters of Care Determination functionality provides the capability for COMS administrators to specify parameters for reports supporting patterns of care determination. Report functionality

foster overview of the provision of oncology services by provider, type of cancer, chemotherapy agent, and other considerations as specified in this panel.

Lab Reports

Functionality for Lab Reports enables COMS administrators to configure laboratory report lists at the local facility that are relevant to the provision of oncology services. Specifically, authorized users will be able to identify the universe of laboratory reports within the local VistA instance and enable availability of the reports for selection as relevant to specific treatment regimens during the template authority process.

4.8.15. Messages Tab

COMS utilizes the "Messages" tab to display any healthcare team messages intended for a specific role ID. Messages are not necessarily intended for a specific user. **Figure 25** shows the "Messages" tab.

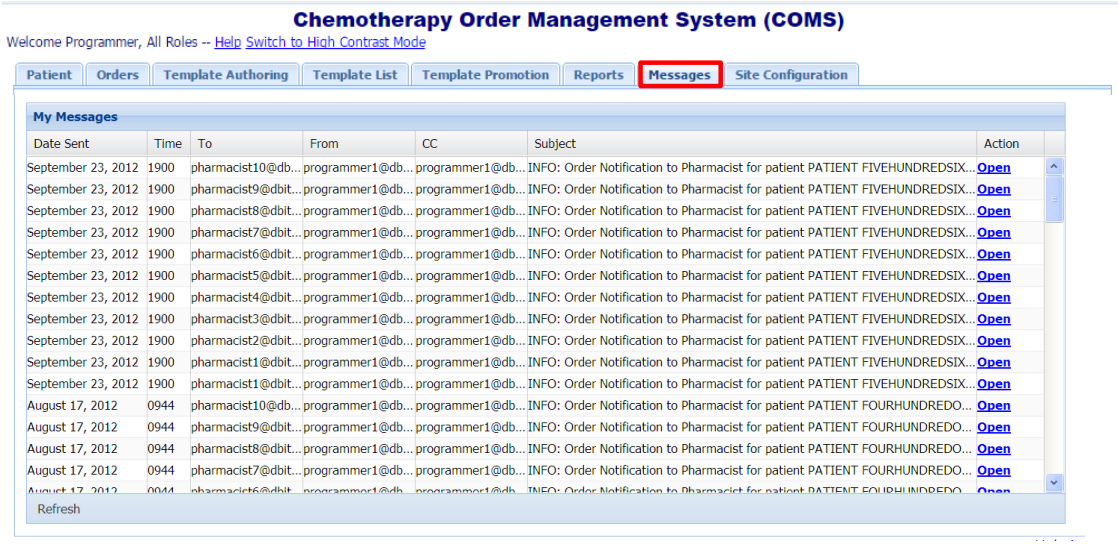

Figure 25: Messages Tab Display

Messages Service Call

The "Messages" service call returns all messages for the role ID listed for the current user. The messages service call is as follows:

- http://example.com/Messages/Filtered/RID/ROLE_ID

COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the following "Message" Model.

```
Message = [              // Array of individual Message JSON Objects
    {
      "mid" : INT,       // Message ID, Integer representing the id of this message
      "MTo" : "",        // Email address of the intended recipient of the message
      "CC" : "",         // Email address of any carbon/courtesy copy recipients
      "Subject": "",     // Subject of the message
```

```
    "Message": "",     // Body of the message
    "Date": {          // Date info of when the message was sent
      "date" : "2012-08-24 11:17:39",
      "timezone_type" : 3,
      "timezone" : "America/New_York"
    },
    "MFrom" : "",      // Email address of the user originating the message action
    "rid" : 26,
    "wid" : 75,
    "dateSent" : "August 24, 2012",
    "timeSent" : "1117",
    "timeZone" : "America/New_York",
    "MStatus" : "Unread",
    "OpenLink" : "https://coms-uat.dbitpro.com/showMessage.php?mid=638"
  },
```

The "Message" data is rendered in the "Messages" tab via a Grid control from the Ext JS library. By default, the data is sorted on the "Date". However, users may change and/or sort data through the Grid's controls.

4.8.16. Site Configuration Tab

COMS provides various administrative functionalities for maintaining and tailoring the application for local facility preferences. Site Configuration is grouped by a series of primary and secondary panels. These are logically aligned from a user standpoint (i.e., how the user would engage this functionality within the application).

Documentation Lists and Contents Panel

The Documentation Lists and Contents panel contains five subordinate panels for authorized users to configure and manage various lists presented throughout the application. **Figure 26** shows the Documentation Lists and Contents panel of the Site Configuration Tab.

Figure 26: Site Configuration – Documentation Lists and Contents Panel

Clinic Information

The clinic information panel enables management of detailed information regarding local facility contacts that is presented as options within the Discharge Instructions panel within the Treatment Documentation module.

Discharge Instructions

The discharge instructions panel facilitates management of detailed information for symptom-based patient instructions presented as options within the Discharge Instructions panel within the Treatment Documentation module.

Lookups

The lookups panel enables authorized users to add, edit, or delete various pull-down menus throughout the application. The "Lookup" service call returns all data for the specified data group as follows:

- http://example.com/LookUp/viewall

COMS returns data in the form of a standard Ext JS Data Store as a JSON object based on the specific "Lookup" Model. The "Lookup" Model for allergies contained within the application is as follows:

```
{
        "id": "29",
        "value": "Allergies",
        "type": "0",
        "description": "Type of Allergy"
    },
```

Medication Documentation

The medication documentation panel enables management of detailed information for medications within a treatment regimen. Within the Discharge Instructions panel of the Treatment Documentation module, this information is presented consistent with medications within the template currently applied to the selected patient.

Toxicity

The toxicity panel enables management of pull-down menu options and associated information for toxicities. Authorized users create toxicity options with Common Toxicity Criteria (CTC) terminology for presentation in the Treatment Documentation module / Assessment panel and Flow Sheet toxicity section of the Add General Information worksheet.

Template Management Panel

The Template Management panel contains five subordinate panels for authorized users to configure and manage various components within the template regimen. **Figure 27** shows the Template Management panel of the Site Configuration Tab.

Figure 27: Site Configuration – Template Management Panel

Delete Template

Functionality for Delete Template enables authorized users to delete any template (i.e., national, local, or user-defined/my templates) within the application. Once a template is deleted, it will no longer be available for use (i.e., apply to patients) within that COMS instance.

- http://example.com/Lookup/Templates

```
{
   "id": "BFF16C4E-74CB-E111-A078-000C2935B86F",
   "type": "4",
   "name": "PACLITAXEL INJ,CONC  200",
   "description": "NSCLC // Paclitaxel Single Agent",
   "totnum": "4",
   "length": "3",
   "unit": "Weeks",
   "coursenum": "0",
   "emolevel": "Low",
   "fnrisk": "5",
   "version": "1",
   "regimenid": "BBF16C4E-74CB-E111-A078-000C2935B86F"
}
```

- http://example.com/Lookup/Templates/GUID

Use "DELETE" HTTP command to delete the template specified by the passed GUID

Disease Staging

Disease Staging functionality permits authorized users to establish relationships between types of cancer and their various disease stages. This enables the application to present options for selection of disease staging in a clinical contextual manner during the template authoring process.

Emetic Medications

Functionality for Emetic Medications facilitates COMS administrators to specify the clinically based Emetogenic Levels of various chemotherapy medications. Once provided within this panel, information for emetic medications is presented during template authoring for the user to consider in determining the regimen's overall Emetogenic Level. This functionality works in tandem with that for Neutropenia/ Emesis Risks to provide mitigation recommendations for providers applying treatment templates and health care teams administering the medications to control these risks associated with the regimen.

Neutropenia/Emesis Risks

Neutropenia/Emesis Risks functionality fosters control of the options for grade/level of Febrile Neutropenia risk and emesis risks and their associated recommendations, based on American Society of Clinical Oncology (ASCO) and National Comprehensive Cancer Network (NCCN) guidelines. Entries in this panel enable information/recommendation functionality for the chemotherapy/biotherapy headers within the Flow Sheet and Treatment Documentation modules as well as regimen-specific narrative for the Order Entry Management module.

Import/Export Template

Functionality for Import/Export Template enable authorized users to download a chemotherapy regimen template from a central library for local modification and/or use. It also enables authorized users to promote locally created templates for review and consideration for inclusion in the central library. Import actions provide template availability within the Chemotherapy Template Order Source module, Template Authoring Tab, and Template List Tab. Export action does not remove the template from use at the local COMS instance and/or local facility.

User Access Panel

The User Access Panel presents two subordinate panels for COMS administrators to view/release applications of the section locked by any user and to add, edit, or delete users. **Figure 28** shows the User Access panel of the Site Configuration Tab.

Figure 28: Site Configuration – User Access Panel

Lockout

Functionality for Lockout enables COMS administrators to view users who have locked sections of the application and release those sections, as appropriate, for subsequent users to access. While COMS notifies individual users of an initial user who has locked a specific section, only a COMS administrator may unlock the section for availability by subsequent users requiring access to the previously locked section.

User Roles

Functionality for User Roles sets access privileges for individual users, assigns roles, and sets permissions to author templates. COMS administrators may use free text entry to specify the user name, E-mail address, and access code; use the pull-down menu to select the user's role within the application; and check the "Template Authoring" box for those users granted those privileges. Actions in this panel enable role-based access for various COMS users.

Clinical Decision Support Panel

The Clinical Decision Support panel presents two subordinate panels for COMS administrators to manage cumulative dose medication options with their recommended maximum lifetime dosages and configure intelligent data entry for comparison of entered vital signs with previous entries. **Figure 29** shows the Clinical Decision Support panel of the Site Configuration Tab.

Figure 29: Site Configuration – Clinical Decision Support Panel

Cumulative Dose Medications

Functionality for Cumulative Dose Medications sets parameters for lifetime cumulative lifetime dosing notifications and overall capabilities within the application. COMS administrators may select the chemotherapy medication to be tracked and enter the recommended maximum dosing per patient that the application will monitor throughout the patient's lifetime. Entries in this panel enable functionality within the Chemotherapy Template Order Source and Order Entry Management modules as well as the Patient Information Panel's Medication Cumulative Dose Tracking section.

Intelligent Data Entry

Intelligent Data Entry (IDE) functionality enables COMS to assess vital sign entries against the identified IDE parameters for maximum/minimum values and/or variance percentages from previous entries and/or clinical guidelines. This functionality is utilized during entry via the Patient Vitals panel and Treatment Documentation module's General Information.

Facility Preferences Panel

The Facility Preferences panel presents seven subordinate panels for COMS administrators to configure the COMS instance to best support local facility policies and practices for various functionalities. **Figure 30** shows the Facility Preferences panel of the Site Configuration Tab.

Figure 30: Site Configuration – Facility Preferences Panel

Active Workflows

Active Workflow functionality enables COMS administrators to specify application processing for various transactions directly, indirectly, or contingent upon further communication/coordination actions and to specify the required actions. This panel controls Messaging functionality and Orders Tab/Order Entry Management module actions as they relate to coordination of changes to ordered medications.

- http://example.com/Admin/ActiveWorkflows


```
{
   "ID": 13,
   "WorkFlowID": 13,
   "WorkFlowName": "Cancelled",
   "Active": 1,
   "Reason": "Notification",
   "NoSteps": 1,
   "ReasonNo": 14,
   "LastIssued": 46,
   "Body": "Your order has been cancelled."
},
```

IV Fluid Types

Functionality for IV Fluid Types enables local facilities to "lock down" specified intravenous (IV) medications to specific fluid types (e.g., normal saline, ringer's lactate). This tailoring of IV Fluid Types implements local facility preferences for the IV medications specified for the COMS instance and directly affects options available within Chemotherapy Template Order Source module, Template Authoring Tab, and subsequently Order Entry Management module.

Medication Holds

Functionality for Medication Holds provides toggle capability to enable the local facility to utilize medication holds or disallow the action/functionality. Tailoring of Medication Holds directly influences the displayed options for Administration Days listed within the Order Entry Management module. Selection of "Yes" enables functionality for Hold/Release from Hold while selection of "No" disables this functionality and does not provide the options.

Medications Not Rounded

Functionality for Medications Not Rounded complements that for Rounding Rules. Specifically, if a facility enables rounding rules of 5% or 10%, COMS administrators may further specify medications that are not permitted to be rounded (i.e., exact dosing for the specified medications). Unless a medication is identified as "not rounded", calculations for the medication will comply with the facility's rounding rules specified in that panel.

- http://example.com/Admin/MedsNonRounded


```
{
  "success": true,
  "total": 1,
  "records": [
    {
      "Lookup_ID": "94A0B573-A48C-E111-A87B-000C2935B86F",
      "Name": "ABACAVIR SOLN,ORAL",
      "NonRounding": 1
    }
  ]
}
```


Rounding Rules

Functionality for Rounding Rules enables COMS administrators to specify whether the local facility permits no rounding, 5% rounding, or 10% rounding. This functionality complements that for Medications Not Rounded. Specifically, if a facility enables rounding rules of 5% or 10%, COMS calculations and users finalizing the medication order may consider rounding to the specified percentage unless identified as "not rounded" in the Medications Not Rounded panel.

- http://example.com/Admin/Rounding


```
{
  "success": true,
  "total": 1,
```

```
  "records": [
    {
      "pctRound": ""
    }
  ]
}
```


Pharmacy Management

Functionality for Pharmacy Management enables pharmacist users with Site Configuration access to synchronize medications lists between COMS and the associated VistA instance. This functionality aligns identical medication lists for the two applications and enables interoperability of COMS with the local VistA instance pharmacy packages. When the COMS application is initially setup, an authorized user must synchronize medication lists to permit interoperability with the associated VistA instance. Periodic synchronization whenever updates are processed for the local VistA instance will ensure the two applications contain identical medication lists and preserve full interoperability of pharmacy packages

Signature Verifications

Functionality for Signature Verifications enables COMS administrators to specify whether the local facility requires one or two healthcare professionals to verify medication dosing prior to administration. This functionality directly impacts the number of users required to "sign to verify" medication dosing on the Treatment Documentation module's General Information panel.