Streamlining Target Fabrication Requests at the National Ignition Facility

ICALEPCS 2017

October 10, 2017

Carla Manin, George Norman, Essex Bond, Raelyn Clark, Allan Casey

NIF Shot Data Systems

LLNL-PRES-739370

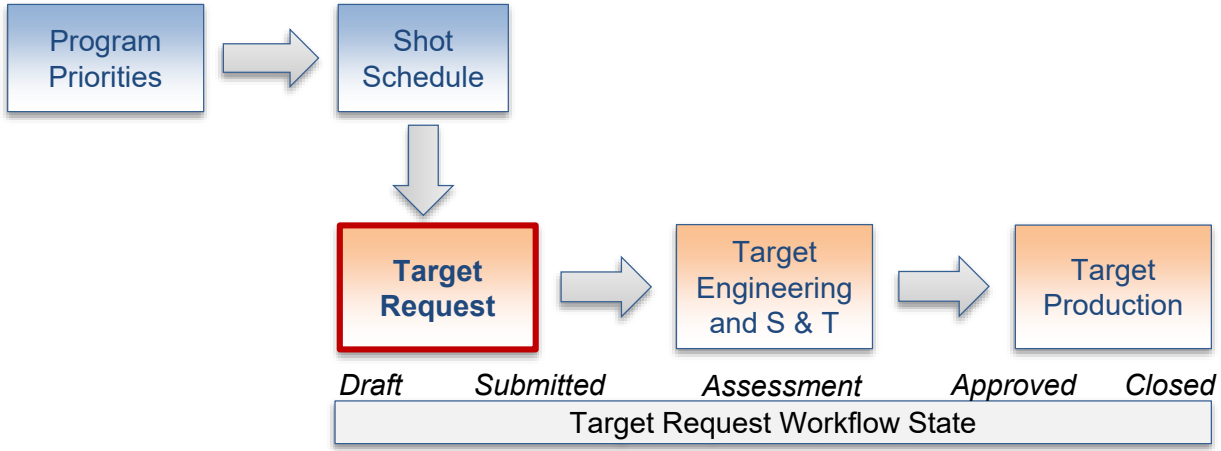
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

NIF Target Fabrication Background

- Estimated 500 targets produced each year.
- Targets are usually standard but many need to be customized.
- Each target produced is paired with an experiment and a request.
- Requests go through a series of approvals before a target is fabricated.

Example of a target

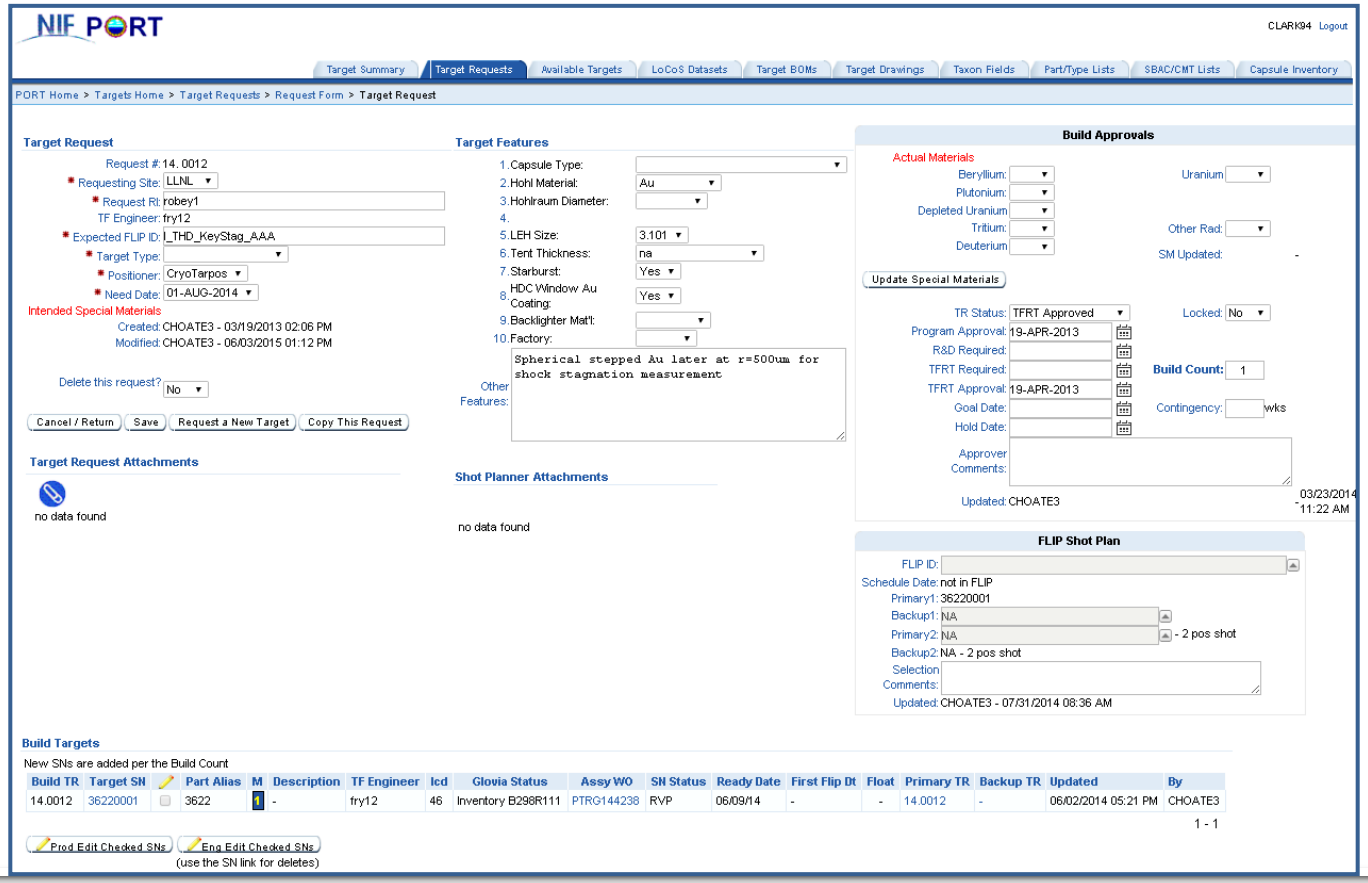


Technician at target fabrication facility

A software application has been used to manage the target requests.

Former Target Request tool

- Display of duplicate data
- Hard-coded features prevent readily adapting to new engineering processes
- Overly complicated logic for sourcing data that slowed the page load time



Limitations of former Target Request tool

- Former target request application developed in Oracle Application Express (APEX) had multiple limitations.

- Process and user interaction limitations:
  - Engineering processes evolved making existing user interaction out of date.
  - Underlying data architecture was not optimized for current use.
  - Development over time led to inefficient use of data.
  - Page loading times were very slow.

- Technology limitations:
  - Customization was possible within the limitations of the APEX framework.
  - All development had to be done through the APEX Web interface.
  - There was no built-in version control.

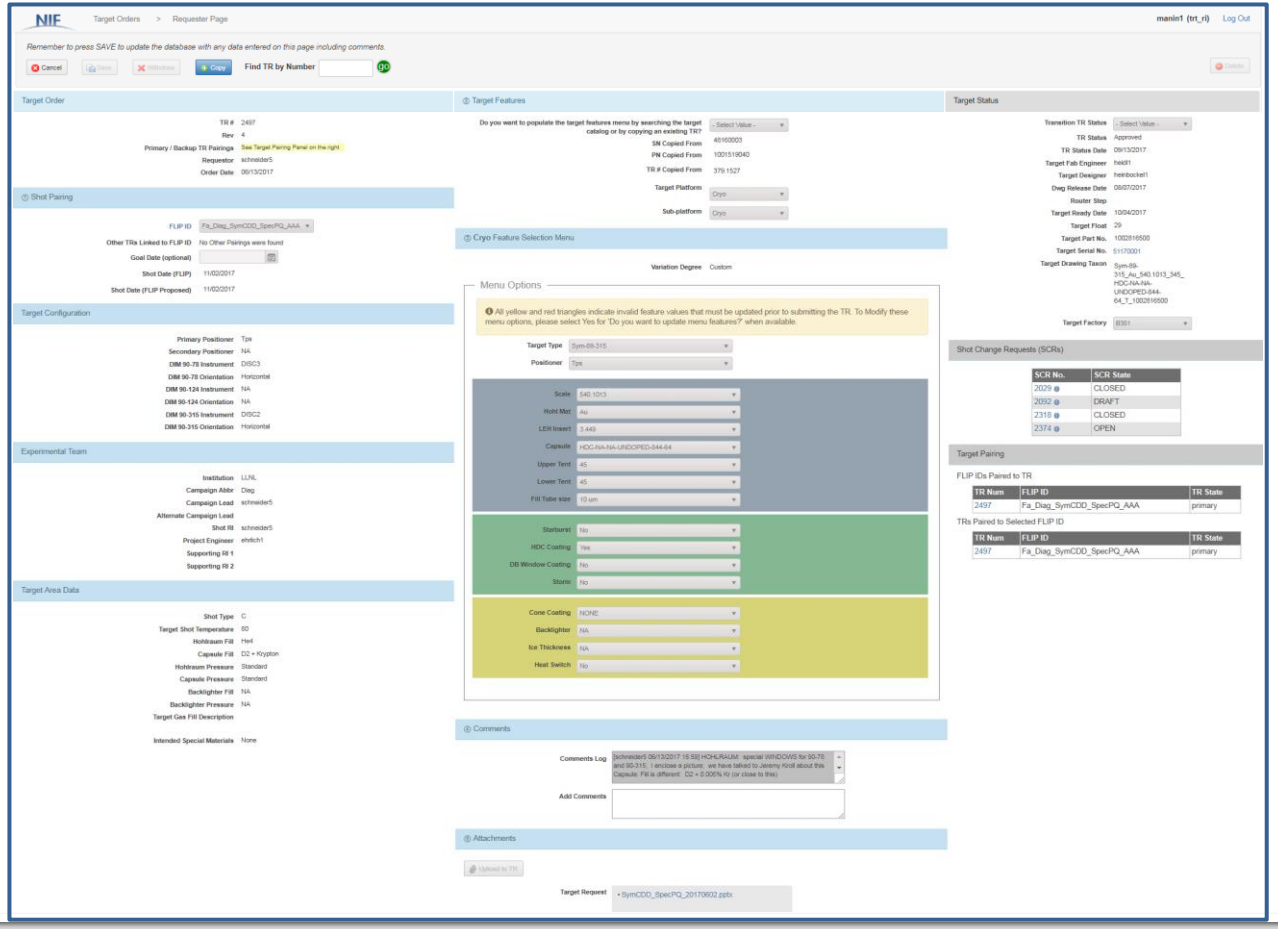
- Decision was made by the Shot Data Systems (SDS) team to develop a new application, with new technologies and on a new data model, rather than modifying the existing one.

New Target Request Tool (TRT)

Display of common data

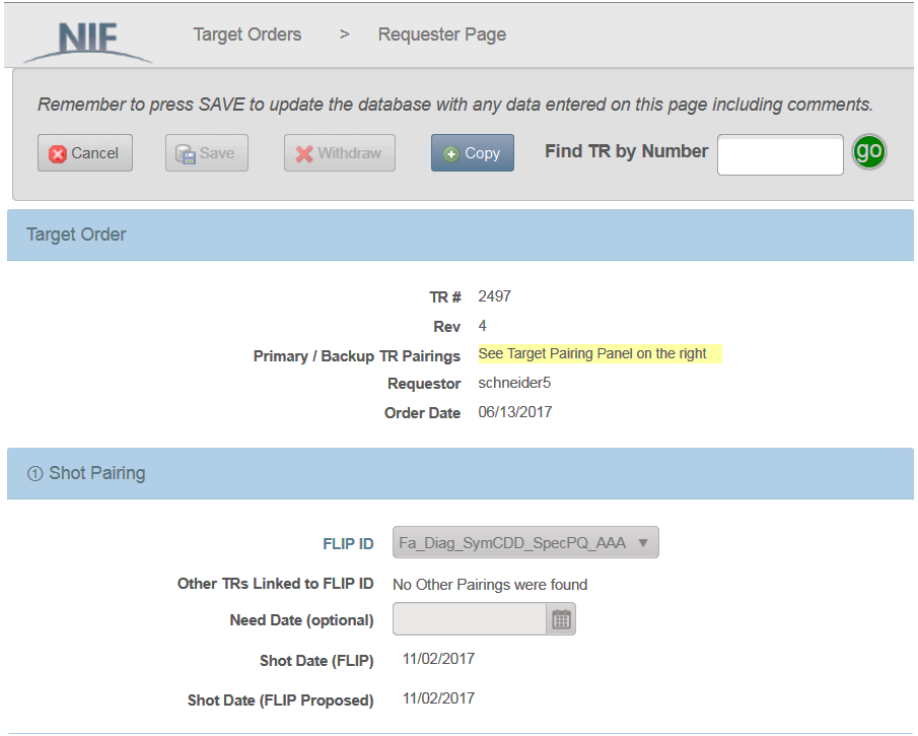
Customizable features that allow for readily adapting to new engineering processes

Simplified logic for sourcing data that improves page load time



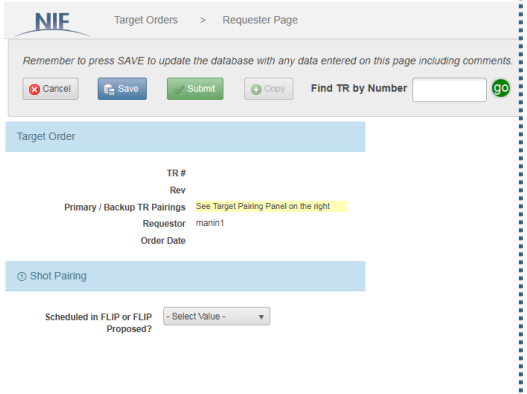
Displayed here is the full view of TRT composed of 3 columns that are gradually displayed as the user makes his/her selections

Zoom in view of two top panels of TRT's left column

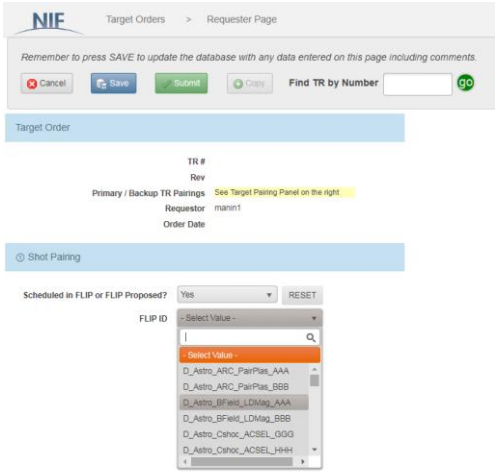


Sequence of steps to create a target request

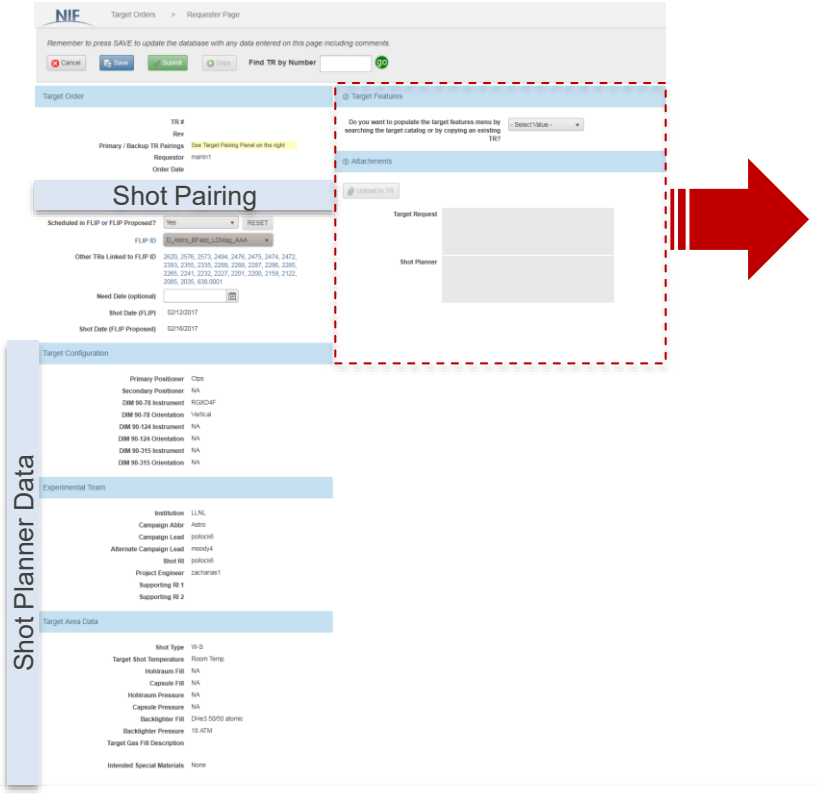
1 New target request
Left panel is displayed



2 Select FLIP* ID



3 Data related to FLIP ID is populated



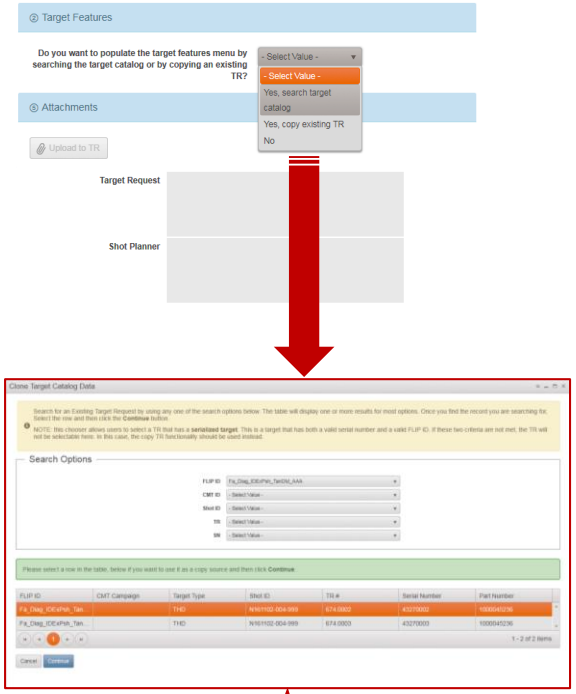
*FLIP = Facility and Laser Integrated Planning

Sequence of steps to create a target request (continued)

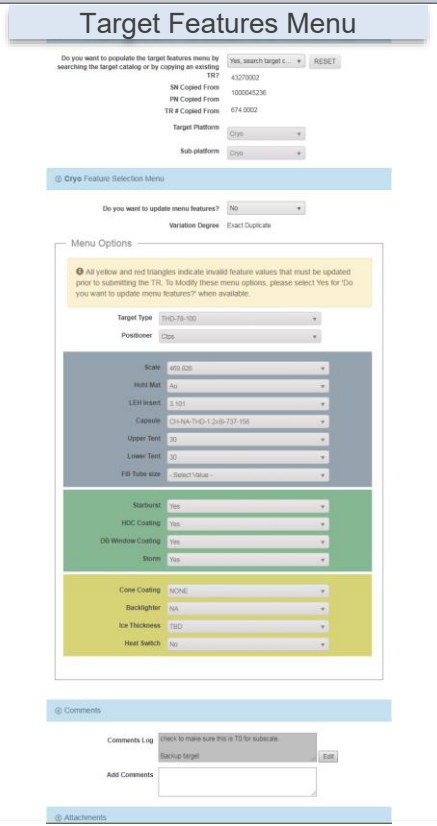
4 Select how to obtain target features

5 Target features are loaded into the middle column

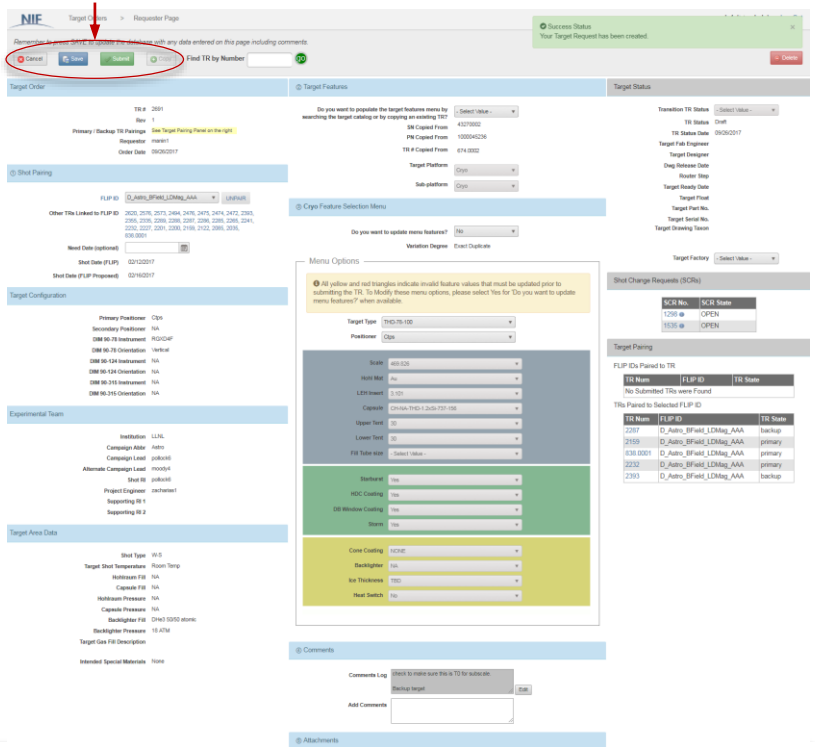
6 TR can be saved. TR# is generated



Pop-up menu to select record to copy from



Action buttons (save, submit, withdraw, cancel)



Technologies chosen for the development of TRT

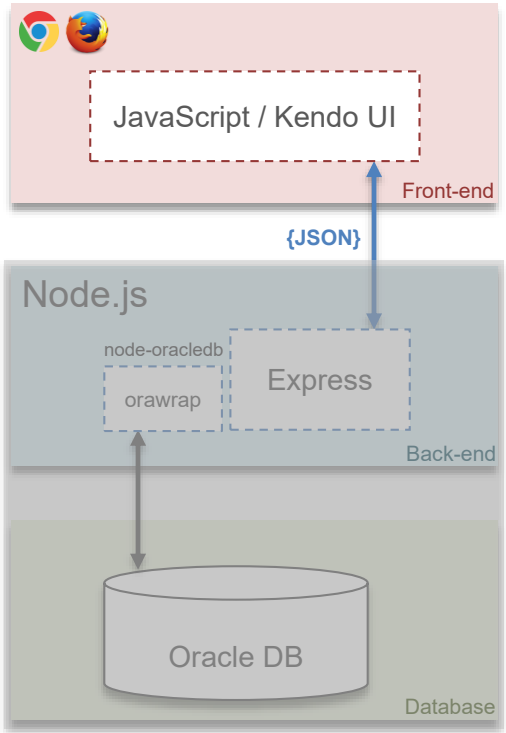
- Node.js
  - Open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side.
  - Modern technology that is supported by a large community of developers.
  - Suitable for non-CPU-intensive operations.
- Express
  - Open-source, minimal and flexible Node.js Web application framework written in JavaScript.
  - *De-facto* standard framework for the majority of Node.js applications.
- Kendo UI
  - Commercial off-the-shelf library for data-rich Web applications that provides more than 70 reusable UI components.
- JavaScript, jQuery, HTML, and CSS
  - Commonly-used Web technologies that allow for an easy implementation with Node.js as the back-end.

Back-end

Front-end

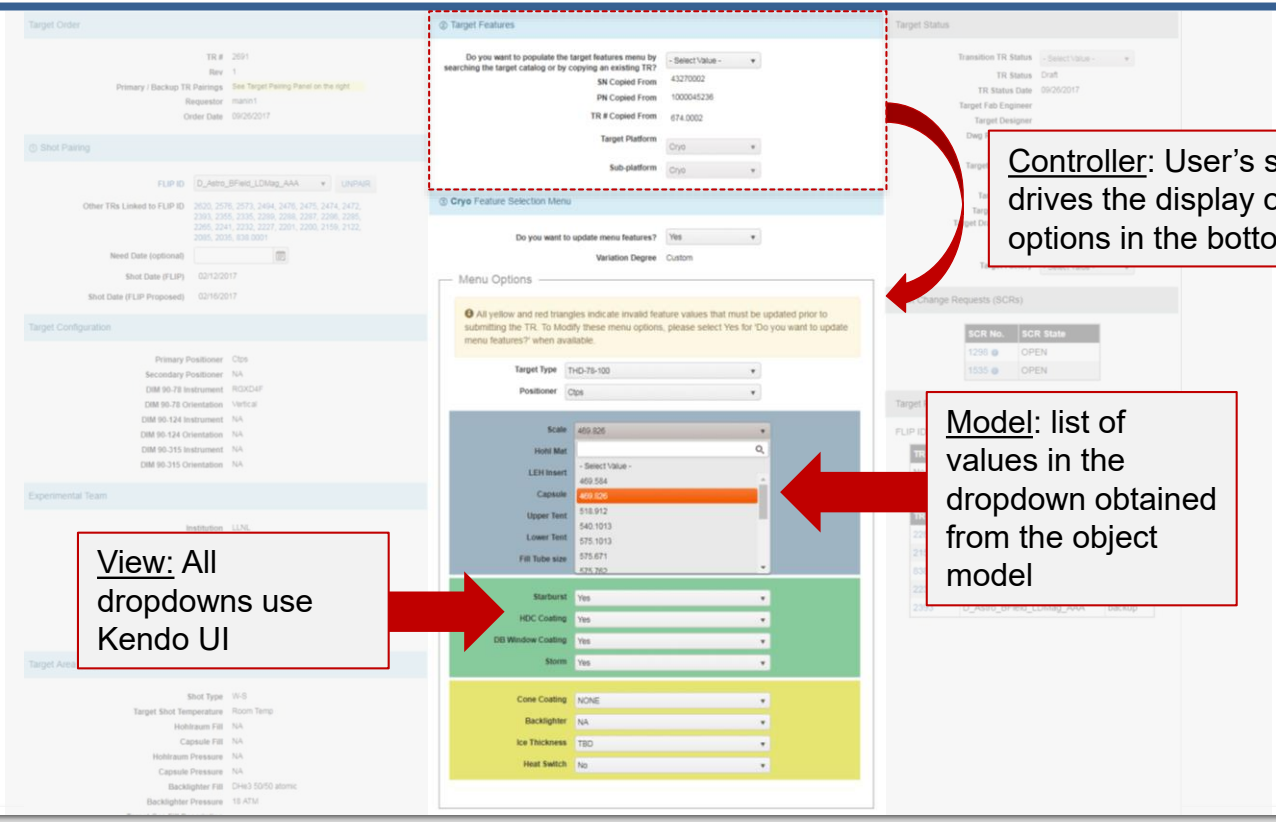
Reasons for using chosen technologies in TRT Architecture

- Front-end:
  - Model-view-controller pattern - selected to provide a clear separation between view and logic.
    - Allowed to easily subdivide the UI into multiple sections and panels which in turn provided flexibility to divide the work among developers.
    - Improved performance significantly as it allowed asking the back-end for data only when needed.
    - Simplified the addition of new panels and features to the UI as most panels are independent and do not need to be reworked for accommodating the new panels.
  - Kendo UI – selected to simplify implementation and speed development.
    - Allowed to customize available UI components to the tool's needs.
    - Provided an easy integration with other Web technologies used in the tool.
    - Improved the look-and-feel of the UI with a simple and clean look.
  - JavaScript, jQuery, HTML and CSS – selected technologies to complement the JavaScript-based back-end.



Developed in a standard model-view-controller pattern.

Example of MVC and Kendo UI in a TRT panel



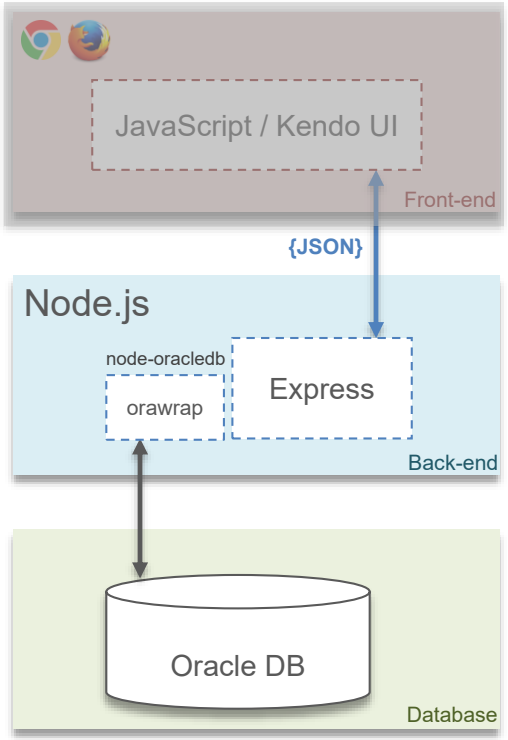
Controller: User's selection drives the display of menu options in the bottom panel

Model: list of values in the dropdown obtained from the object model

View: All dropdowns use Kendo UI

Reasons for using chosen technologies in TRT Architecture (continued)

- Back-end:
  - Node.js – has proven to work well and be reliable for other SDS applications.
    - Allowed us to seamlessly connect to existing Oracle database.
    - Provided a fast turnaround for developing the application.
    - Paired well with Web technologies used for the front-end.
  - Express – is easy to use and provides a well-written online documentation.
    - Provided multiple methods for querying the request and constructing the result as a JSON file.
    - Provided a thin layer of fundamental Web application features, without obscuring Node.js features.
- Database:
  - Oracle database – is the supported infrastructure for the facility and used for all SDS applications.
  - 'node-oracledb' driver - manages a fast and stable database connection.
  - 'orawrap' library - creates a listening pool on the provided port and provides an easy way to handle SQL queries.
    - *The orawrap library is no longer being maintained. It has been added to the core Oracle database driver (node-oracledb).*



Conclusion

- The use of modern technologies allowed the SDS team to meet the overall project goals primarily within the development time allocated.
- TRT provides faster loading time, improved user interaction, and smooth data integration.
- Future maintenance is simplified given the MVC pattern adopted.

Thank you


Lawrence Livermore
National Laboratory