SAPVis: Visualizing a SAP Network

CS547 Project Proposal

Vaden Masrani
vadmas@gmail.com

November 9, 2015

1 Domain Description and Task Analysis

SAP (Systems, Applications & Products) is software used by large multinational corporations to manage their daily operations. SAP systems manage inventory, sales, expenses, payroll, and everything else organizations need to keep track of in order to run efficiently. These system are typically very large and interconnected and because they are often managed by changing staff, contain many inefficiencies including duplicate or unused code, ineffective or outdated test suites, expired users and redundant data tables. A tool that can help technicians quickly locate these inefficiencies would help companies improve their SAP system and save them money.

The dataset consists of item and link data where each item has categorical, ordinal and quantitative attributes. The main categorical attributes are type (eg. user, program, transaction code, function, class, table, view), domain (eg. "PS - Project System") and id (eg."PROG-ZZ_CONVERT_SDCC_DATA"). The ordinal attributes are creation and last-used dates. The main quantitative attributes are performance, usage, size and degree. The performance and usage attributes are scores between 0 and 100, the size is either the size in bytes or by number of lines, and the degree is an integer which measures the number of incoming and outgoing edges. Some attributes are null (empty strings) for some items.

Link data shows the relationships between various items. A link between $item\_i$ and $item\_j$ means $item\_i$ calls or has access to $item\_i$ in the SAP system. This can be a user who has permissions to modify a particular program, or a program which uses a particular data table. The links are directed; a user can have permission to a modify a program but a program can't modify a user. Circular calls, where program A calls program B which calls program A are possible (a potential source of inefficiency) and are represented by a directed cycle in the graph.

Because SAP software systems are often large and interconnected, SAP technicians need a tool to allow them to consume (rather than produce) their data in order to find inefficiencies in the network that are costing the company money. Although SAP systems often consist of more than 100 000 nodes, the networks can be split into "domains" consisting of

1000- 2000 nodes and links. The proposed tool will handle one domain and therefore have to display approx. 2000 nodes and links. The tool should allow users to search and query their network in order to discover outliers (eg. programs that aren't called, users with no permissions, tables that aren't used), find similarities in network topologies (perhaps a source of duplicate and therefore redundant code), find extremum (eg. programs with high usage and low performance,), and see dependancy relationships (in order to direct testing should something be changed upstream in the call chain). The tool should also support the four types of search, "Lookup", "Browse", "Search", "Explore". Use cases for these four search scenarios are outlined below.

2 Proposed Visualization

The proposed visualization, as seen in the attached figure, uses a combination of the *facet, encode, reduce, manipulate* and *focus+context* idioms to join an arc diagram with a tree network. Two views, a focus view and a context view, are facetted together along with a control panel on the left which allows the user to reduce the data with various sliders which select a range from the quantitative attributes. The context view at the bottom is an arc diagram which shows the entire domain grouped by types. Each type is then ordered by one of its attributes in descending order, where the size of the node is encoded with a quantitative attribute of the users choice. Each node is also encoded with unique colour to show membership to a particular group. The dropdown menu above each group allows the user to order the nodes by a particular attribute, and the top n (undecided at time of proposal) nodes are listed underneath the dropdown.

Arcs join the nodes along the top to show connections between nodes (the final vis will have directed arcs unlike in figure). The user can manipulate the context view by zooming in or out along the x axis, and by panning and translating the context view to see other nodes. Although the context view can support displaying all nodes in the network, the user is expected to use the attribute sliders on the left navigation panel to reduce the size dataset while viewing the context view.

The user can select a node either directly from the context view or by entering an ID using the search bar, and that displays the selected node on the focus view. The focus view allows the user to see the dependancy hierarchy for a particular node. In this example, the user has selected the program "PROG-ZP6_NTSKK_LIST_ACTIVITY_V2" from the context view. This displays the node in the focus view and show that user VADMAS has permission to modify "PROG-ZP6_NTSKK_LIST_ACTIVITY_V2" which calls two functions in red. The function "ZP6_NTSKV..." then calls four transaction codes (TCODES). This hierarchical tree view allows the user to quickly see a dependency chain within the domain. The use case for such a chain is discussed below. On the right of the focus view is a dropdown of stats related to that particular selection. The user can lock the statistics to the page so as to quickly compare this selection to future selections.

On the left is a control panel that allows the user to filter the data. The filters can be layered and this panel can be extended to allow control over more quantitative attributes.

In the figure, the user has selected the subset of data with performance between 8% and 20%, and usage between 13% and 71%. As well, there is a global-sort drop down which allows the user to sort all groups by the same attribute. The same effect could be achieved by manually changing each dropdown in the context view, but this global sort is provided for convenience. There is also a set-order selector which allows the user to reorder and hide entire groups. Finally, there is a search bar at the top right side of the vis which allows the user to quickly select a particular item.

This vis will be implemented using D3, jQuery, HTML and CSS and will be supported by Firefox, Chrome, and Safari. I will not be building on any preexisting software or toolkits, but may use small plugins for dropdown widgets and the search bar.

3 Use Case

Best Buy asks a technician to improve the performance of their SAP system, as it is taking too long to scan items and customers are getting upset with long lines at checkout. The technician begins by sorting the types by degree and exploring the data, making a note of all the users, programs, transaction codes and tables have degree 0. These items aren't called by any parts of the SAP system and are likely left over from previous developers. He documents these as being potentially able to be deleted.

Next, using the filters, he reduces the context to only show those items with poor performance and high usage. Browsing the context view, he notices half a dozen programs that are used often but have a low performance score. By selecting each program from the context view, he loads the call chain into the hierarchy view and exports the full set of items that rely on each of these highly-used-but-low-performing programs. Each of these programs may have to be rewritten and if so, the accompanying set of dependent items will need to be tested to assure the rewrites are accurate. By testing only these items rather than the full test suite the technician is able to save Best Buy money and resources.

While exporting these programs, he notices the topology of call chain around "PROG-ZP6_NTSKK_LIST_ACTIVITY_V3" and "PROG-ZP6_NTSKK_LIST_ACTIVITY_V2" look very similar. He thinks it's odd that both V3 and V2 are being used in the same system and realizes that V2 should have been deleted after V3 was created. He suspects their may also be a V1 that can be deleted and uses the search bar to enter "PROG-ZP6_NTSKK_LIST_ACTIVITY_V1" By looking at the topological similarly between V3 and V2 he was able to identify redundant code, V2 and V1, and improve the performance of the network.

4 Personal Expertise and Milestones

I have little experience using D3 and slightly more experience using JavaScript and doing front end work. My previous programming experience has been with imperative, object-oriented languages so D3 is a significant change. I expect the learning curve to be steep

and will be doing many online tutorials.

This visualization will not be building on previous work. There was an earlier version which tried to used a 2D force layout (rather than an arc diagram) for the context view, but we have made significant changes to the layout and therefore will be starting fresh with this V2. In terms of domain experience, I have very little but I'm working closely with a company (CodeExcellence) which is providing the data and can answer domain questions. Because this work will be continued after the course is completed and eventually turn into a commercial product, the code must be robust and production-ready.

I believe it's a little ambitious to try to implement the entire visualization by December 15th, especially because the code needs to be robust and commercial grade. Therefore, I plan to focus on the context view and control panel for this course. My milestones are listed below.

Milestone	Date	Target
1	Nov 15th	Begin learning D3, do online tutorials, implement a few toy D3 charts
2	Nov 22th	Implement a static arc diagram
3	Nov 23th	Status Update
4	Nov 29th	Add mouse handlers, sorting animation, tooltip on hover, highlight family on click
5	Dec 6th	Add Pan and Zoom, Dropdown Menus, Sort by attribute
6	Dec 15th	Control panel, allow filter by attribute, animation for adding/removing nodes when filtered, final presentation
7	Dec 18th	Final Paper

5 Previous Work

There has been lots of work on displaying large networks [2]. Among the most popular options are force directed 2D layouts, adjacency matrices, arc diagrams, circular layouts, and more recent approaches such as pivot graphs and bubble tree layouts [2], [5]. Our first attempt was to use a force directed 2D layout approach. Because the network is highly connected, the 2D layout suffered from the "hairball" problem [3]. We could have used a software package like Gephi to help with the layout but that requires manually positioning nodes in a graph which is time consuming and wouldn't be helpful to SAP technicians [1]. Also, running the force layout on thousands of nodes took upwards of 20 seconds which

was too slow. We also considered using an adjacency matrix. A matrix view would solve the performance issues and would be able to scale to thousands of nodes, but it requires significant training in order for network analysts to understand the data that is being displayed [3]. Because we want this tool to be used by non-technical users, we wanted to avoid the steep learning curve associated with adjacency matrices. For similar reasons we avoided the pivot graph and bubble layout approach and opted for a more standard arc layout following the visualization mantra "Overview first, zoom and filter, then details-on-demand." [4], We hope this will solve the performance and scale issues without having a steep learning curve for users.

References

[1] M. Bastian, S. Heymann, and M. Jacomy, *Gephi: An open source software for exploring and manipulating networks*, 2009.

[2] M. J. McGuffin, *Simple algorithms for network visualization: A tutorial*, Tsinghua Science and Technology, 17 (2012), pp. 383–398.

[3] T. Munzner, *Arrange network and trees*, in Visualization Analysis and Design, A. K. Peters, ed., CRC Press, 2014.

[4] B. Shneiderman, *The eyes have it: A task by data type taxonomy for information visualizations*, in Proceedings of the 1996 IEEE Symposium on Visual Languages, VL '96, Washington, DC, USA, 1996, IEEE Computer Society, pp. 336–.

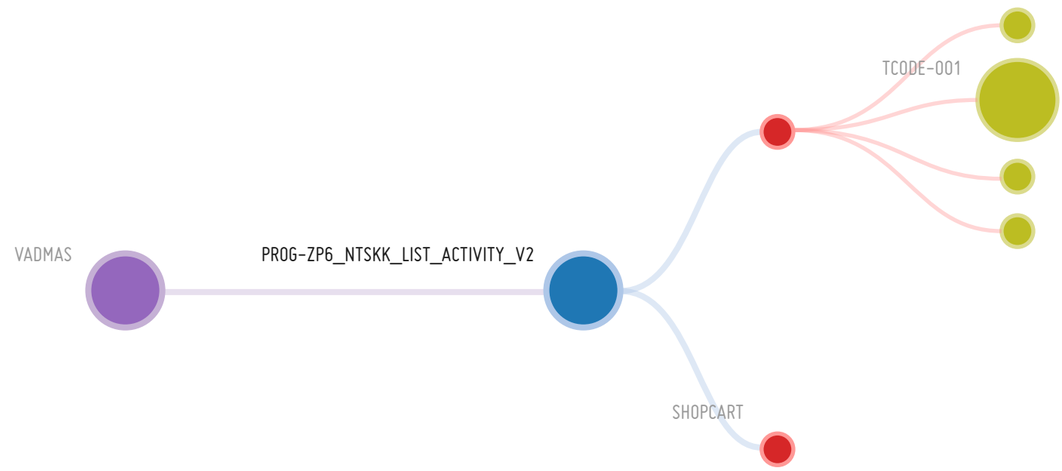
[5] M. Wattenberg, *Visual exploration of multivariate graphs*, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '06, New York, NY, USA, 2006, ACM, pp. 811–819.

SEARCH PROG-ZP6_NTSKK_LIST_ACTIVITY_V2

SET ORDER USERS
GLOBAL SORT BY CONNECTIONS

DEGREE 0 10
USAGE 13 71
PERFORMANCE 8 20

SCALE BY PERFORMANCE



PROG-ZP6_NTSKK_LIST_ACTIVITY_V2

DEGREE
USAGE
PERFORMANCE
CREATE DATE
SIZE
LINES OF CODE

TCODES
TABLES
USERS
FUNC
METHODS
INCL

EXPORT CSV

PROG-ZP6_NTSKK_LIST_USERS

SORT BY CONNECTIONS

ZP6_NTSKV...	87%
LIST_ACTIVIT...	83%
TELECOM...	79%
TOOLFORM-X...	75%
SHOPCART-3G...	71%
ACTUAL-HOUR...	67%

SORT BY DATE CREATED

ZP6_NTSKV...	2015-01-05
LIST_ACTIVIT...	2015-01-06
TELECOM...	2015-01-07
TOOLFORM-X...	2015-01-08
SHOPCART-3G...	2015-01-09
ACTUAL-HOUR...	2015-01-10
EQUIP_00...	2015-01-11
SHOPCART-3G...	2015-01-12
ACTUAL-HOUR...	2015-01-13

SORT BY SIZE

ZP6_NTSKV...	2000 LOC
LIST_ACTIVIT..	1900 LOC
TELECOM-...	1800 LOC
TOOLFORM-X...	1700 LOC
SHOPCART-3G...	1600 LOC

SORT BY USAGE

ZP6_NTSKV...	87%
LIST_ACTIVIT...	83%
TELECOM...	79%
TOOLFORM-X...	75%
SHOPCART-3G...	71%
ACTUAL-HOUR...	67%

SORT BY DATE CREATED

ZP6_NTSKV...	2015-01-05
LIST_ACTIVIT...	2015-01-06
TELECOM...	2015-01-07
TOOLFORM-X...	2015-01-08
SHOPCART-3G...	2015-01-09
ACTUAL-HOUR...	2015-01-10
EQUIP_00...	2015-01-11
SHOPCART-3G...	2015-01-12
ACTUAL-HOUR...	2015-01-13