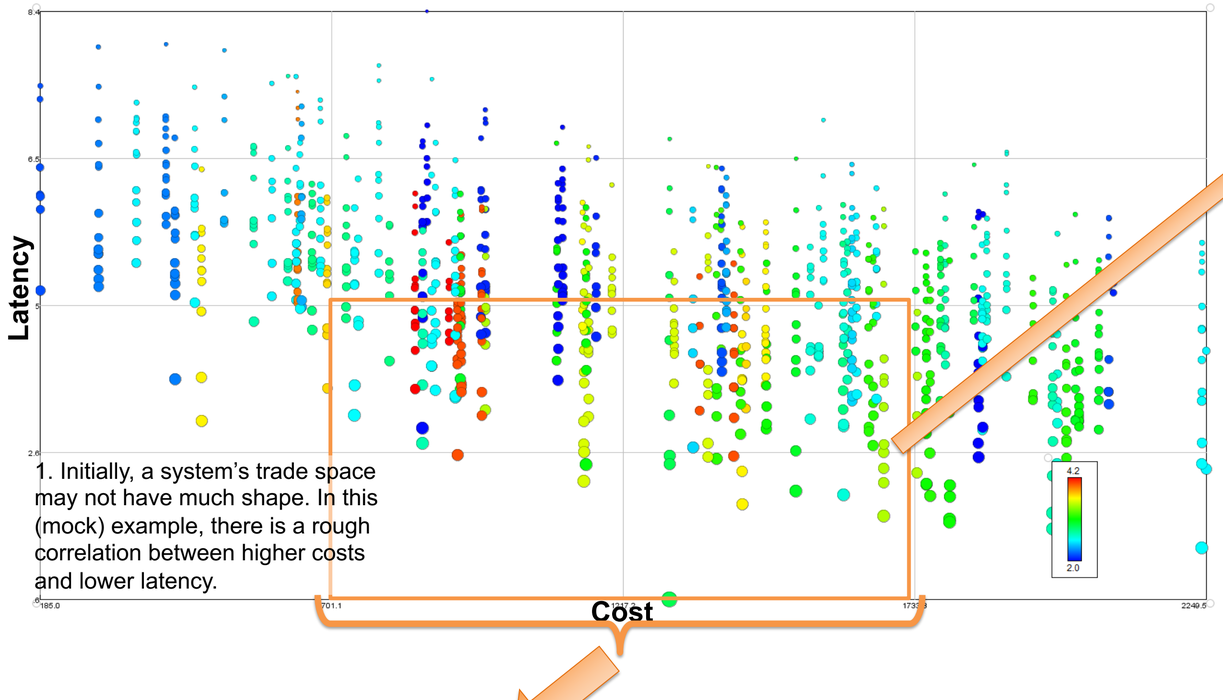
Guided Architecture Trade Space Exploration of Safety Critical Software Systems

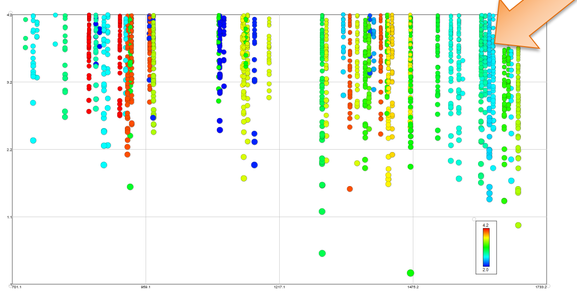
Modern systems are so heavily interconnected that the impacts of component and configuration choices can be difficult to know at design time. In this work, we partially automate the exploration of a system's *architectural trade space* to enable system designers to rapidly evaluate system design options. The result is a graphical, user-guided suite of tools that enables a 'design-by-shopping' style of system design.

We build on SEI's existing work in high-fidelity system architecture modeling. Over the past several years, SEI has developed the *Architecture Analysis and Design Language* (AADL), which enables the creation of very detailed models of a system's architecture. The SEI has also developed the *Open Source Architecture Tool Environment* (OSATE), which contains a number of analyses for AADL models, ranging from weight and latency calculations to more sophisticated error-behavior modeling.

A system's *architectural trade space* is the set of all possible candidate designs plotted in n-dimensional space where each dimension is a different quality attribute. Penn State's *ARL Trade Space Visualizer* (ATSV) enables designers to visually explore multidimensional and potentially infinite data sets, and to focus in on particular subsets. It uses an evolutionary algorithm to learn which input characteristics correspond to which output characteristics; this lets designers refine their searches and generate additional candidate architectures that meet particular criteria.



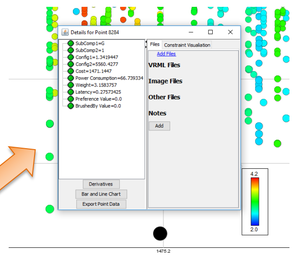
1. Initially, a system's trade space may not have much shape. In this (mock) example, there is a rough correlation between higher costs and lower latency.



2. Designers can refine their searches, and ATSV will generate more candidate architectures in the reduced search area. This is done automatically, using an evolutionary algorithm to learn which input values correspond to which outputs.

This work has three primary tasks:

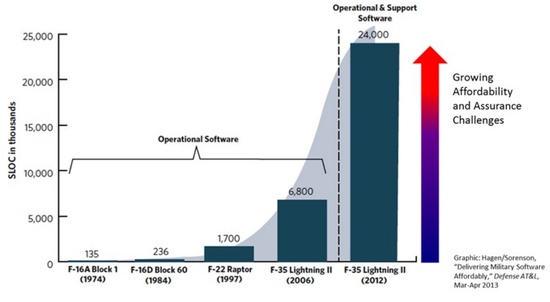
1. Extend existing architecture modeling language (SEI's AADL) to encode component choices and their interactions.
2. Extend existing architecture modeling tooling (SEI's OSATE) to automatically analyze the resulting system for cost, weight, performance, etc.
3. Enable trade space visualizer (Penn State's ATSV) to automatically select valid components and configurations, and visually display analysis results.



3. Once a satisfactory architecture has been identified, the designer can click the point to reveal the component and configuration options chosen, as well as exact quality attribute values for each dimension.

How does 'design-by-shopping' work? Much like traditional shopping, design-by-shopping begins with a broad search of *candidates* that potentially meet some need. Individual candidates are evaluated according to attributes like cost, performance, weight, power, etc. If none of the candidates are satisfactory, more can be generated – or the trade space can be refined to focus the search. If one of the candidates is satisfactory, though, then the designer can view its exact configuration and proceed to more detailed modeling and analysis.

Why do we need something new? System development costs are rising at an unsustainable rate. Much of this rise is driven by software, which presents a number of unique challenges in terms of massive customizability/configurability and subtle interactions with other components. SEI's work in system architecture modeling addresses this need head-on by modeling software, hardware, and the bindings between the two.



Software as % of total system cost
1997: 45% ⟶ 2010: 66% ⟶ 2024: 88%

This work is an enabling technology for future architecture modeling research at the SEI. Any quantitative analysis can be used as a dimension of the trade space visualization. Sophisticated new analyses for AADL are being created in OSATE to analyze complex system properties like safety and security. Once built, it's relatively straightforward to automate these analyses and connect them to ATSV.

This project combines a number of technologies to enable a new paradigm of system design. As it relies heavily on quantitative analyses of system architectures, in the future we will look into novel quantification strategies for traditionally qualitative attributes like safety and security.

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Contact: Sam Procter sprocter@sei.cmu.edu

Copyright 2017 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific entity, product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute nor of Carnegie Mellon University - Software Engineering Institute by any such named or represented entity.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License. Requests for permission for non-licensed uses should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM24-1383