Dynamic Data Structures
Future Plans and Work

DDS is an ongoing research project at Rhode Island College. The system continues to grow and evolve. This document presents some of the future plans for DDS.

Classroom

Look for collaborators.

Evaluate effectiveness as a teaching aid for instructors.

Develop labs and exercises for DDS and evaluate its effectiveness as a student learning aid.

Provide opportunities for undergraduate research projects.

Software

All changes should maintain an easy to use interface for the basic modeling capabilities. The tool should remain easy to use when teaching. The instructor should be concentrating on their teaching, and not worrying about using DDS!

Rebuild the internal state machine. Capture actions on selected data in the selection class.

Document! Document! Document! Document!

Complete implementation of the ~~binary tree and~~ doubly linked list version~~s~~.

Incorporate a tabbed interface. This will make it easier to organize classroom demos as well as assign and collect student assignments.

Implement full screen mode. Who doesn't want more real estate for their project?

Implement an undo facility.

Add option to drag entire sub-lists or entire sub-trees of a selected node.

Add a capability to neatly layout a list or tree.

Improve placement of new nodes and reference variables. All node reference variables are placed at same location.

Design and implement a record and play animation feature. This will allow instructors to build demonstrations for classroom usage so they do not have to manually manipulate the data structures. It will allow students to record their solutions to problems given in an assignment.

Design and implement a feature to model recursion. This is needed especially for binary tree problems. Need to keep the user interface simple. Should start with a push/pop capability on Node Reference variables. A full implementation may need to model the use of local variables, parameter passing, and a return value.

Add node coloring.

Add a highlighting facility.

Add ability to remove Node Reference variables.

For C programmers, add ability to delete a node.

Copy and paste.

Multiple selection and drag.

Under Consideration

Allow an option to use string or integer payloads.

Allow an option to use key-value payloads.

Add optional arrowheads.

Add labels.

Add pop-up annotations.

Should there be a way to model objects, such as a List object that contains a head, tail, and curr pointer. Then the user can model multiple lists by allocating these objects. Or can we accomplish this by allowing Node Reference variables to use dotted names, then rely just on the names.

Allow parent references on tree nodes. Some algorithms use parent references instead of recursion. They can simplify moving up and down the tree.

Long Term

A block language feature has been prototyped. Consider a full implementation.

Output tikz/pgf models for use in TeX document.

Implement variables of same type as payload with assignment of the data between variables and node payload.

Interface with Canvas.

2-3 trees, 2-3-4 trees, B-trees?