Modularity/Reuse

- Make soar more object oriented
  - Build on visual soar data map
    - Public vs Private data
    - Type safety (maybe flag for runtime checks)
  - Allow definition of methods
    - enforce separate naming conventions
    - auto renaming/refactoring
  - Automatically generate html/pdf documents
    - Coupling with SoarDoc

Debugging

- Tracing
  - breakpoints
  - undo command
  - “load complete: 3 syntax errors, 4 overwritten rules”
- Profiling
  - operator-level profiling
    - why is this operator so expensive
  - automatic experimentation to evaluate different multi-attribute settings

- Debugging
  - Diff to debug in Visual C++/Visual Studio
  - Visual Soar is not debugging tool
  - Need improved Syntax checking in Soar and Visual Soar
  - SOTA: TSI (Tcl-Soar Interface), SoarDB,
  - Other systems' solutions/solutions: visual cues re I-supported/O-supported
  - breakpoints
  - tracing and undo
  - graphical view of working memory?
  - better UI to existing functionality
  - see memory changes in more dynamic way
- Better support of Code reuse
  - How to develop structured data types shared among modules
  - How to encapsulate to grab and reuse (partly a documentation issue)
  - Datamap has comment fields…integrate SoarDoc and Visual Soar using VisualStudio.Net (/// for headers)
  - SOTA: data maps, conventions,
  - Other systems' solutions/solutions: OOP, private/public, name spaces/auto renaming mechanism, automated refactoring
- How to make it easy to learn new tools
- Representational compactness
- More formal mechanisms for doing macros
  - (but macro languages make it harder to debug)
  - SOTA: write lots of rules
  - Rx: more compact macros facility
- Legacy code & examples…how to find it, know it's available
- Repository of info…to help new code generation
- Need "load complete with no errors" message
- Profiling…
  - Would like to know "within this operator [what] is chewing up all this time"
  - Flag re changing rules how predicates are evaluated…multi-attributes command…computing matches is one cost; computing how good a match is, adds a huge computational burden
- "Undo" capability
- Save Soar states
- Automated coverage testing
- Hard to stop Soar using SGIO
- How to get Soar to communicate with other architectures
- System can't be learned without understanding theory behind it all…fundamental differences between Soar and any other coding aren't readily grasped from the canned tutorial, example code.
- Are there tutorials for the tools?
- Hard to learn on line…couldn't figure out how to use Visual Soar.